W3School Linux 教程

wizardforcel

Published
with GitBook



目錄

W3School Linux 教程

来源：Linux 教程

整理：飞龙

Linux 基础

Linux 简介

Linux内核最初只是由芬兰人李纳斯·托瓦兹（Linus Torvalds）在赫尔辛基大学上学时出于个人爱好而编写的。

Linux是一套免费使用和自由传播的类Unix操作系统，是一个基于POSIX和UNIX的多用户、多任务、支持多线程和多CPU的操作系统。

Linux能运行主要的UNIX工具软件、应用程序和网络协议。它支持32位和64位硬件。Linux继承了Unix以网络为核心的设计思想，是一个性能稳定的多用户网络操作系统。

Linux的发行版

Linux的发行版说简单点就是将Linux内核与应用软件做一个打包。

目前市面上较知名的发行版有：Ubuntu、RedHat、CentOS、Debain、Fedora、SuSE、OpenSUSE、TurboLinux、BluePoint、RedFlag、Xterm、SlackWare等。

Linux应用领域

今天各种场合都有使用各种Linux发行版，从嵌入式设备到超级计算机，并且在服务器领域确定了地位，通常服务器使用LAMP（Linux + Apache + MySQL + PHP）或LNMP（Linux + Nginx+ MySQL + PHP）组合。

目前Linux不仅在家庭与企业中使用，并且在政府中也很受欢迎。

- 巴西联邦政府由于支持Linux而世界闻名。
- 有新闻报道俄罗斯军队自己制造的Linux发布版的，做为G.H.ost项目已经取得成果.
- 印度的Kerala联邦计划在向全联邦的高中推广使用Linux。
- 中华人民共和国为取得技术独立，在龙芯过程中排他性地使用Linux。
- 在西班牙的一些地区开发了自己的Linux发布版，并且在政府与教育领域广泛使用，如Extremadura地区的gnuLinEx和Andalusia地区的Guadalinex。
- 葡萄牙同样使用自己的Linux发布版Caixa Mágica，用于Magalh?es笔记本电脑和e-escola政府软件。
- 法国和德国同样开始逐步采用Linux。

Linux vs Window

目前国内Linux更多的是应用于服务器上，而桌面操作系统更多使用的是Window。主要区别如下：

比较	Windows	Linux
界面	界面统一，外壳程序固定所有Windows程序菜单几乎一致，快捷键也几乎相同	图形界面风格依发布版不同而不同，可能互不兼容。GNU/Linux的终端机是从UNIX传承下来，基本命令和操作方法也几乎一致。
驱动程序	驱动程序丰富，版本更新频繁。默认安装程序里面一般包含有该版本发布时流行的硬件驱动程序，之后所出的新硬件驱动依赖于硬件厂商提供。对于一些老硬件，如果没有了原配的驱动有时很难支持。另外，有时硬件厂商未提供所需版本的Windows下的驱动，也会比较头痛。	由志愿者开发，由Linux核心开发小组发布，很多硬件厂商基于版权考虑并未提供驱动程序，尽管多数无需手动安装，但是涉及安装则相对复杂，使得新用户面对驱动程序问题（是否存在和安装方法）会一筹莫展。但是在开源开发模式下，许多老硬件尽管在Windows下很难支持的也容易找到驱动。HP、Intel、AMD等硬件厂商逐步不同程度支持开源驱动，问题正在得到缓解。
使用	使用比较简单，容易入门。图形化界面对没有计算机背景知识的用户使用十分有利。	图形界面使用简单，容易入门。文字界面，需要学习才能掌握。
学习	系统构造复杂、变化频繁，且知识、技能淘汰快，深入学习困难。	系统构造简单、稳定，且知识、技能传承性好，深入学习相对容易。
软件	每一种特定功能可能都需要商业软件的支持，需要购买相应的授权。	大部分软件都可以自由获取，同样功能的软件选择较少。

Linux 安装

本章节我们将为大家介绍Linux的安装。

本章节以 centos6.4 为例。

centos6.4 下载地址：

- 网易镜像：http://mirrors.163.com/centos/6/isos/
- 搜狐镜像：http://mirrors.sohu.com/centos/6/isos/

注：建议安装64位Linux系统。

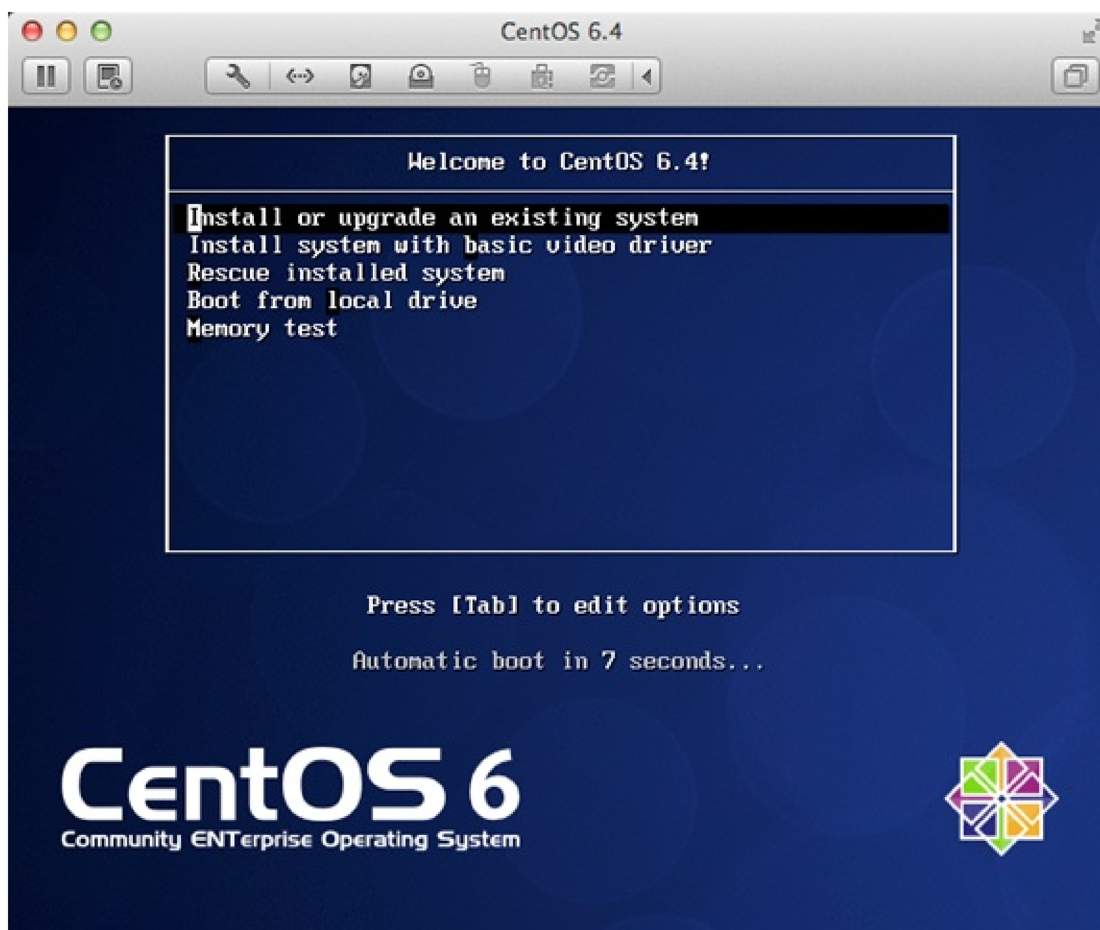
接下来你需要将下载的Linux系统刻录成光盘或U盘。

注：你也可以在Window上安装VMware虚拟机来安装Linux系统。

Linux 安装步骤

1、首先，使用光驱或U盘或你下载的Linux ISO文件进行安装。

界面说明：



Install or upgrade an existing system 安装或升级现有的系统

install system with basic video driver 安装过程中采用基本的显卡驱动

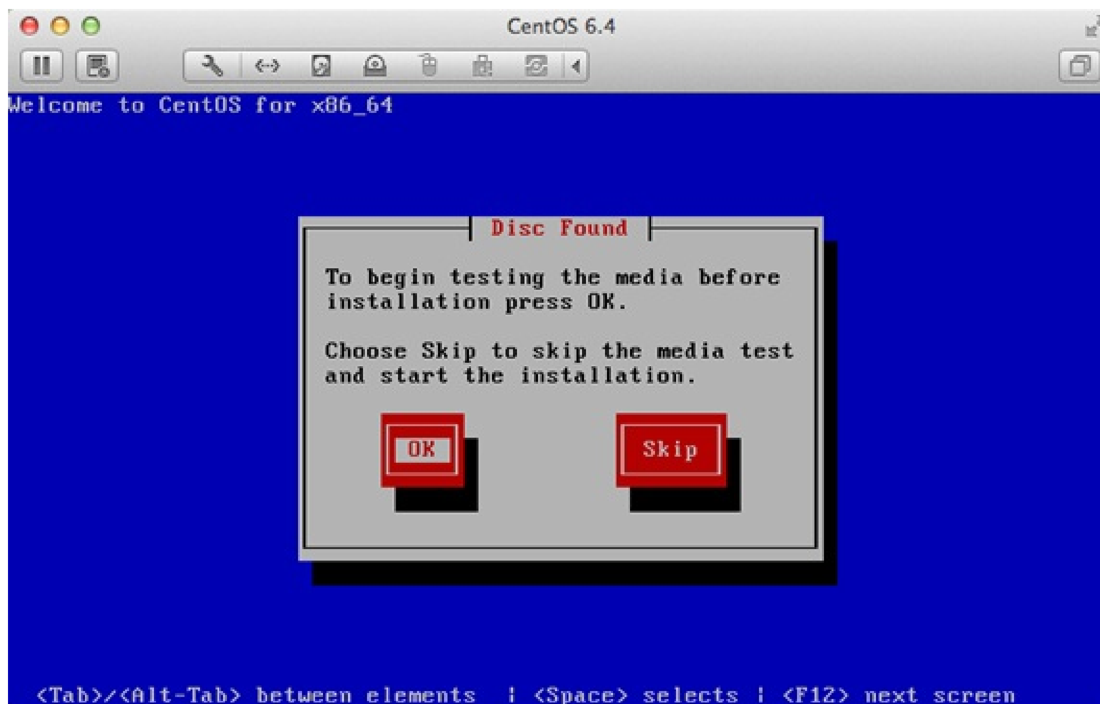
Rescue installed system 进入系统修复模式

Boot from local drive 退出安装从硬盘启动

Memory test 内存检测

注：用联想E49安装时选择第一项安装时会出现屏幕显示异常的问题，后改用第二项安装时就没有出现问题

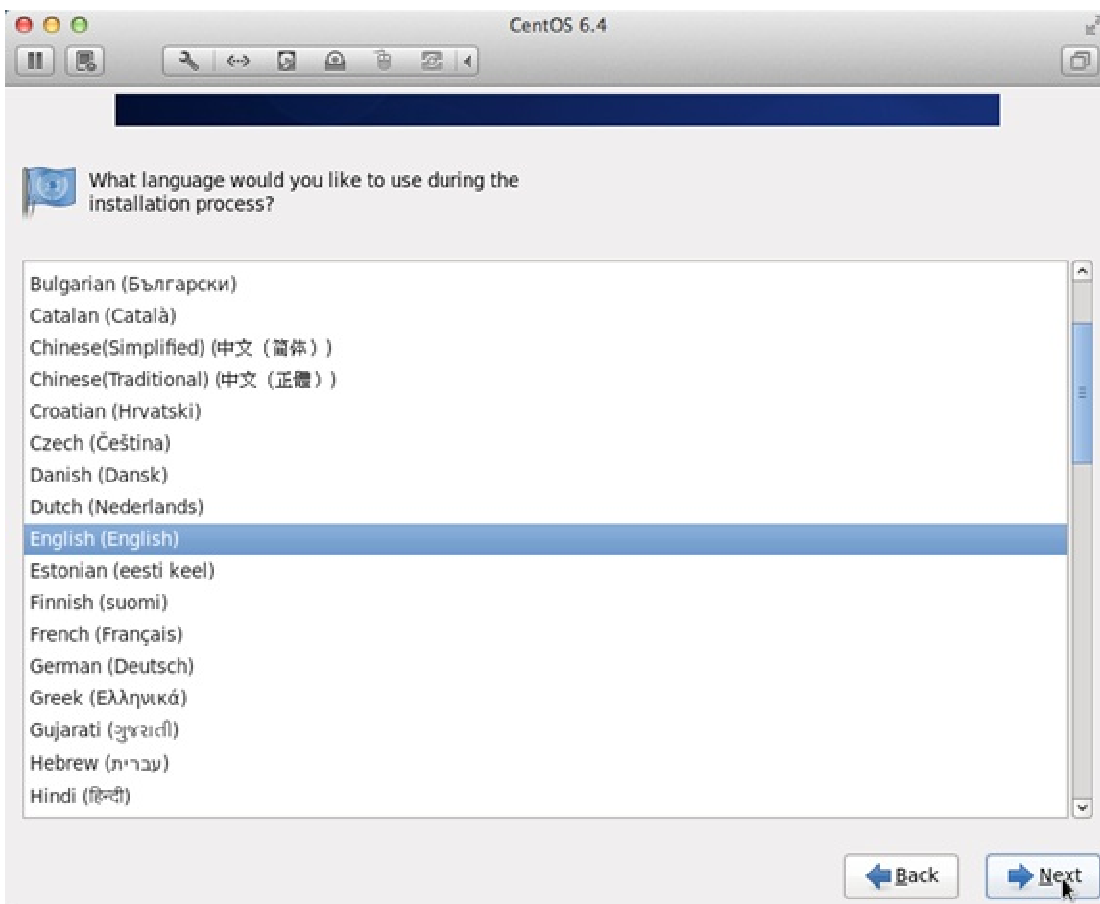
2、介质直接"skip"就可以了



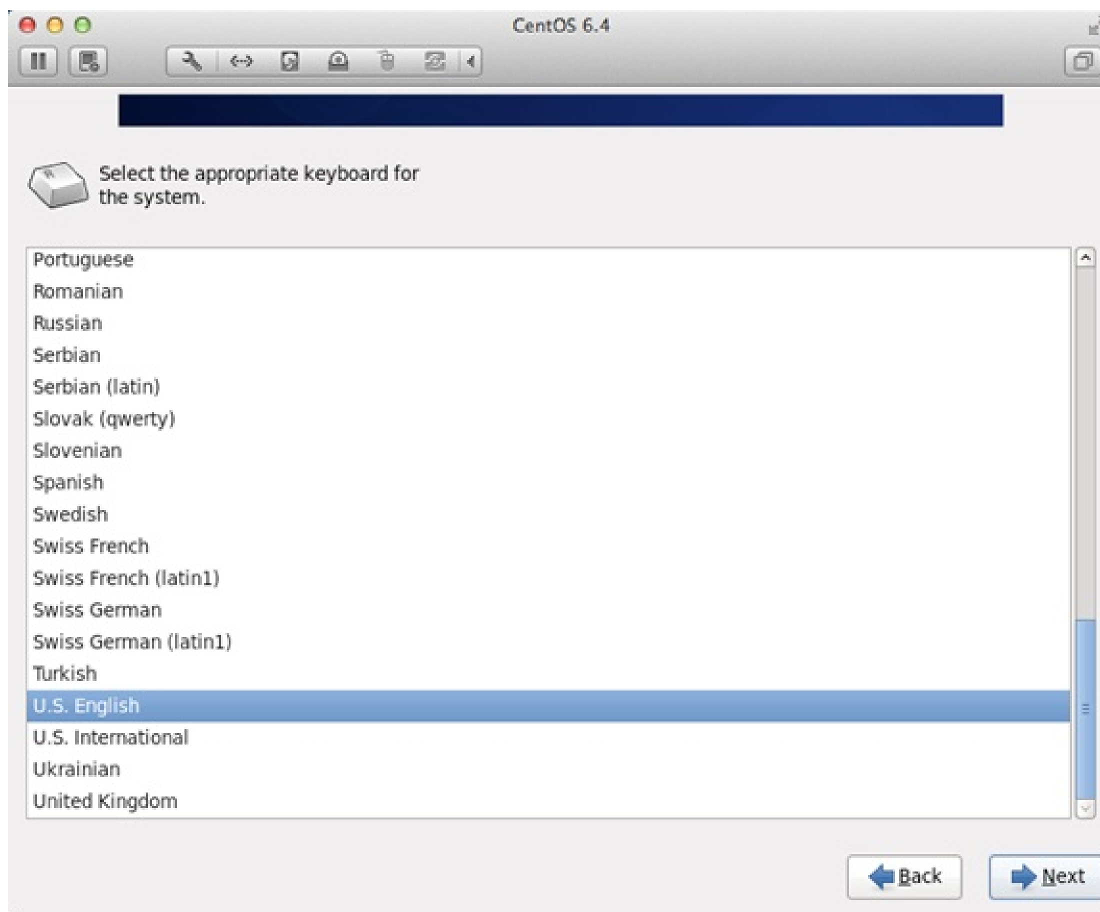
3、出现引导界面，点击"next"



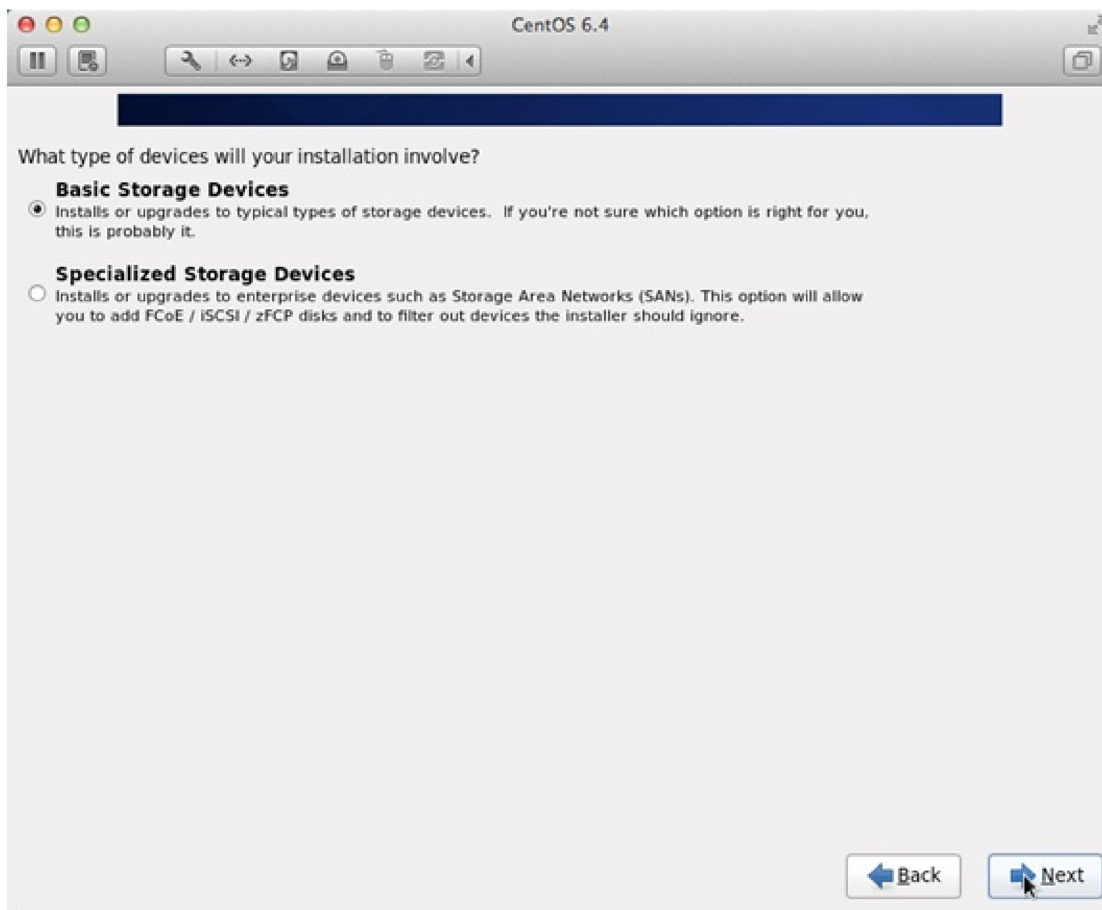
4、选中"English（English）"否则会有部分乱码问题



5、键盘布局选择"U.S.English"



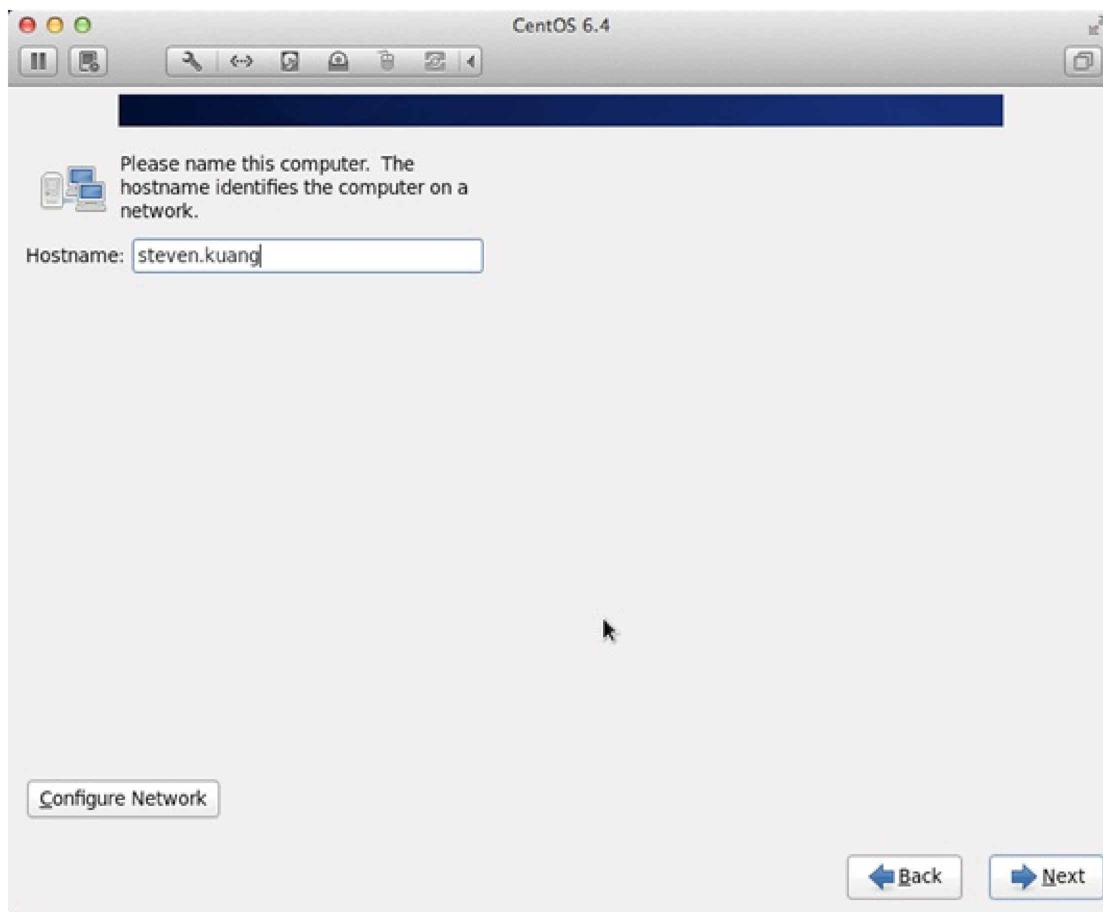
6、选择"Basic Storage Devies"点击"Next"



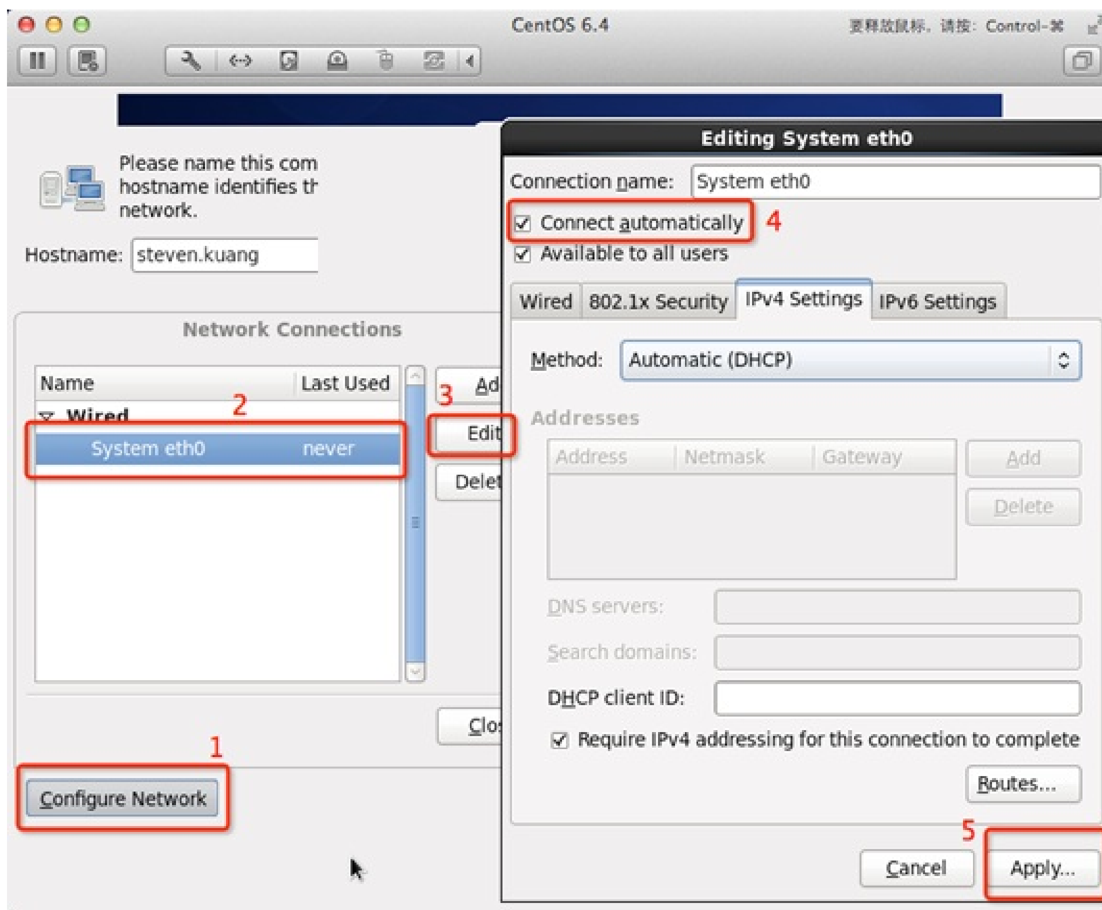
7、询问是否忽略所有数据，新电脑安装系统选择"Yes,discard any data"



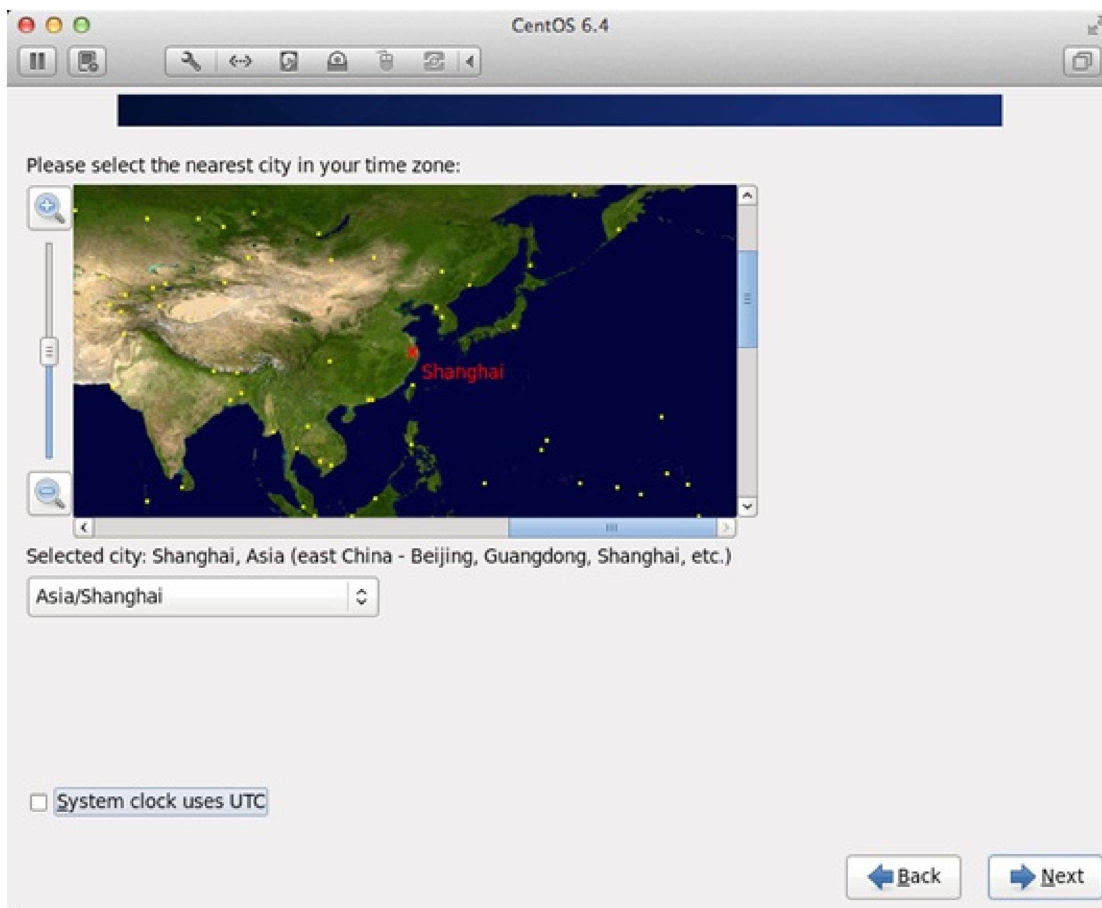
8、Hostname填写格式"英文名.姓"



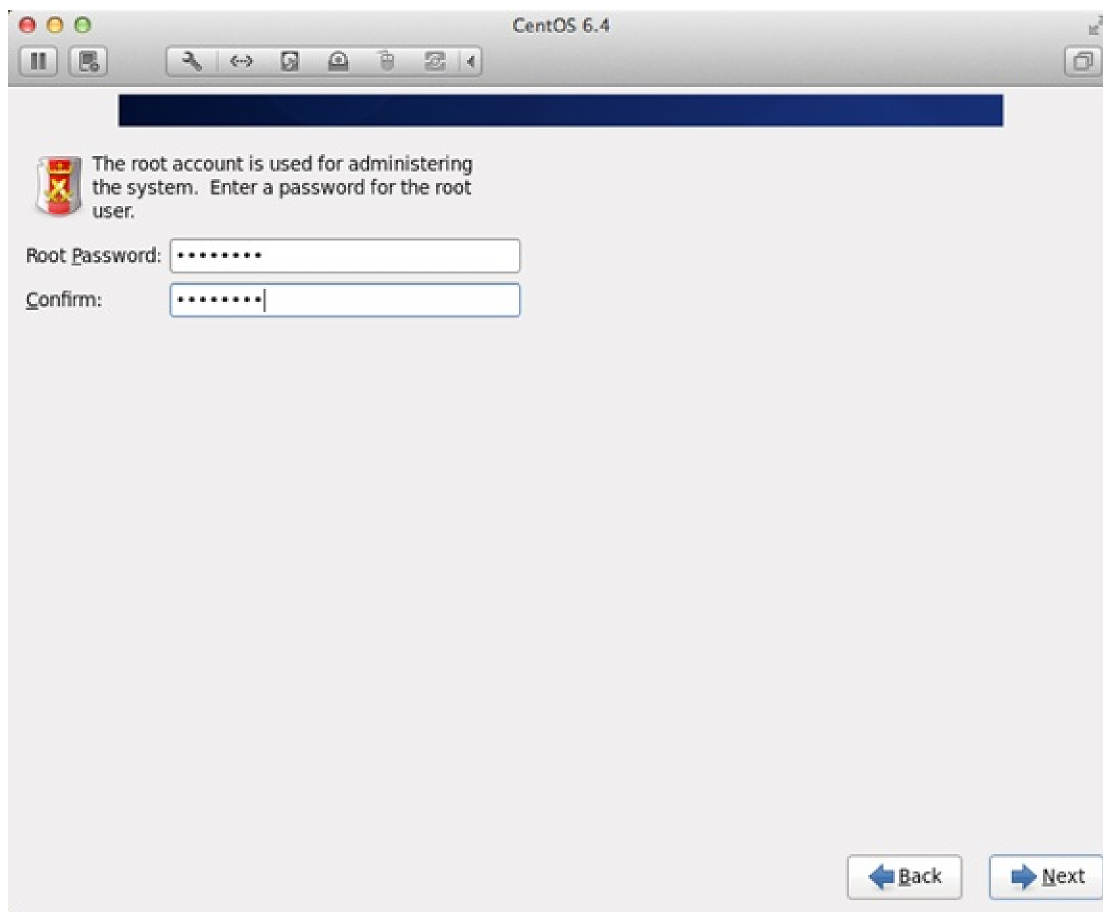
9、网络设置安装图示顺序点击就可以了



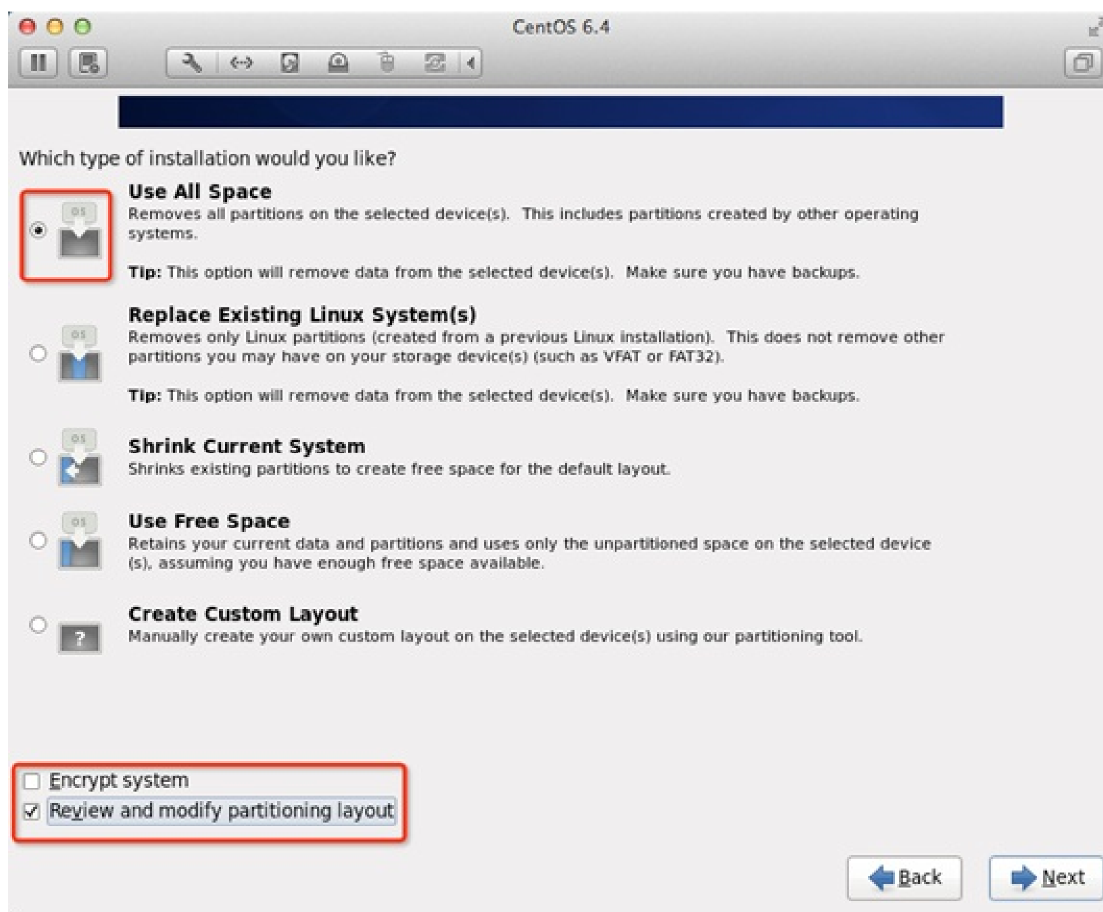
10、时区可以在地图上点击，选择"shanghai"并取消System clock uses UTC前面的对勾



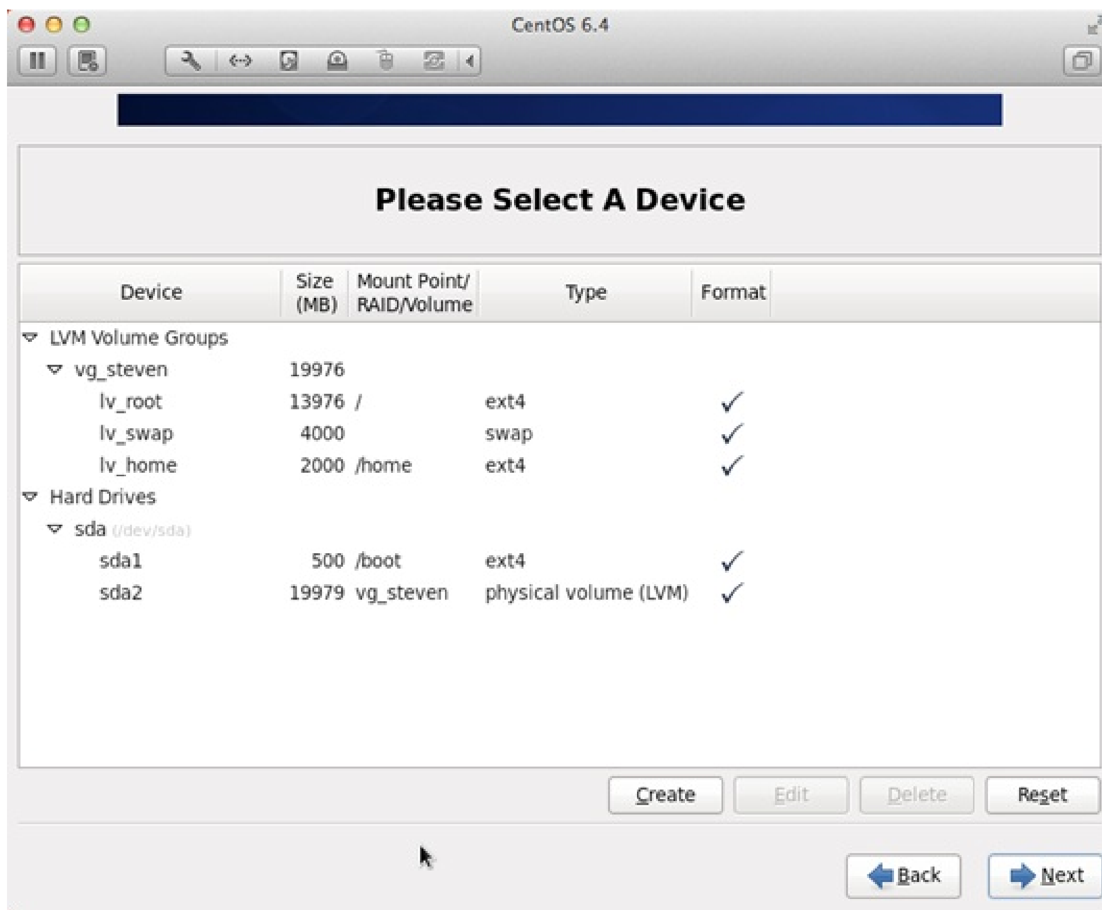
11、设置root的密码



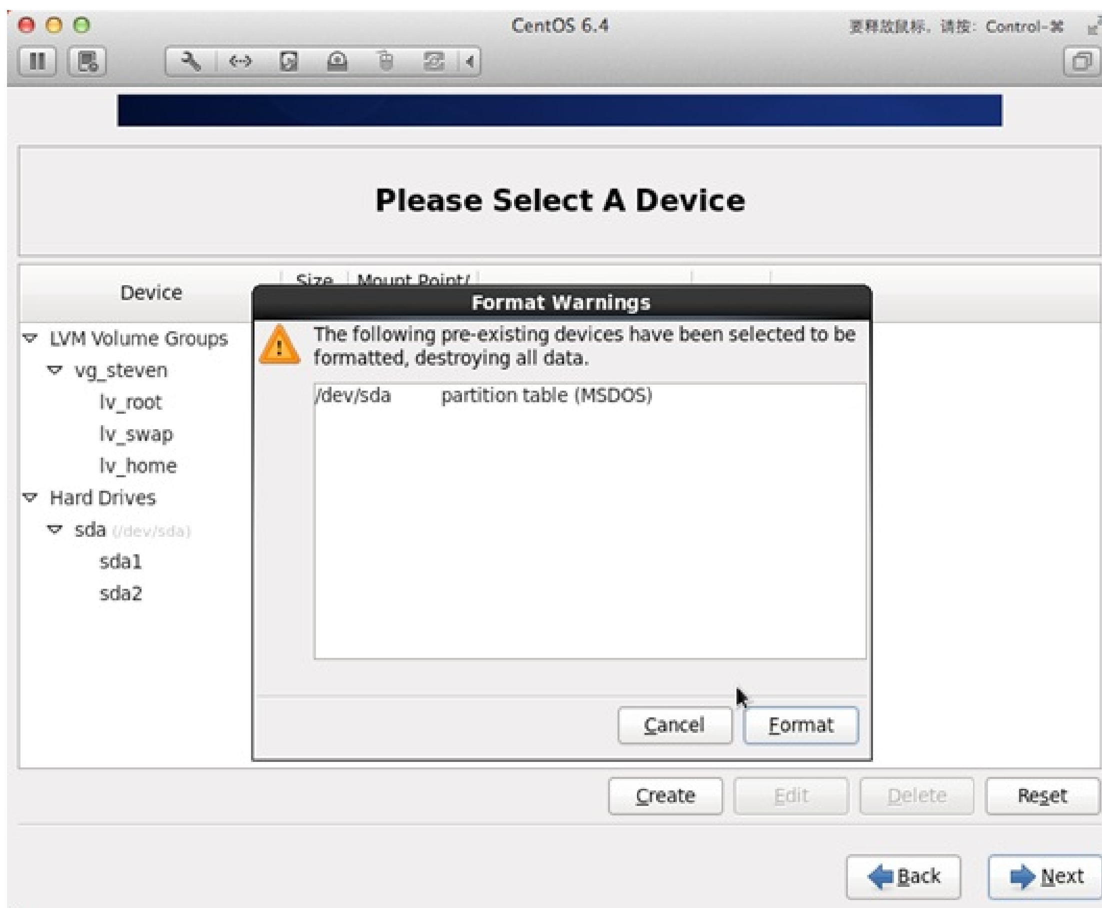
12、硬盘分区，一定要按照图示点选



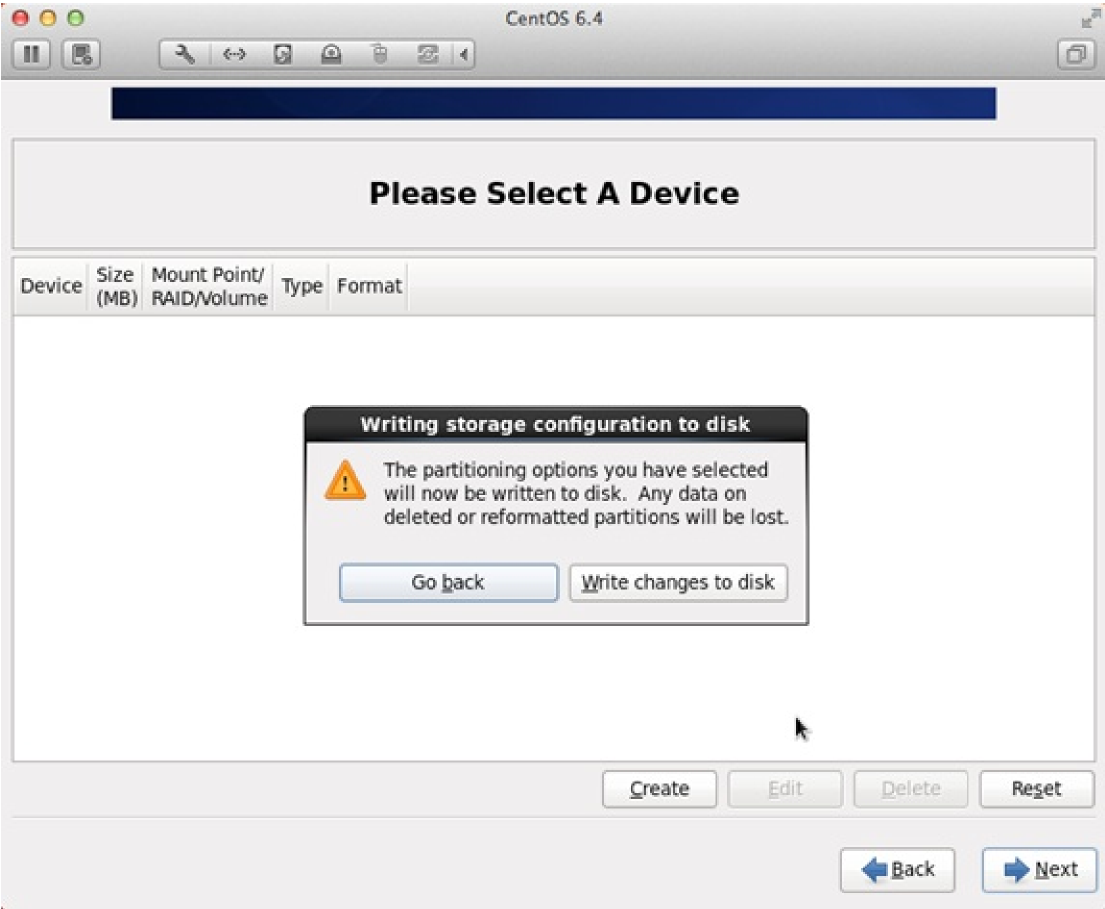
13、调整分区，必须要有/home这个分区，如果没有这个分区，安装部分软件会出现不能安装的问题



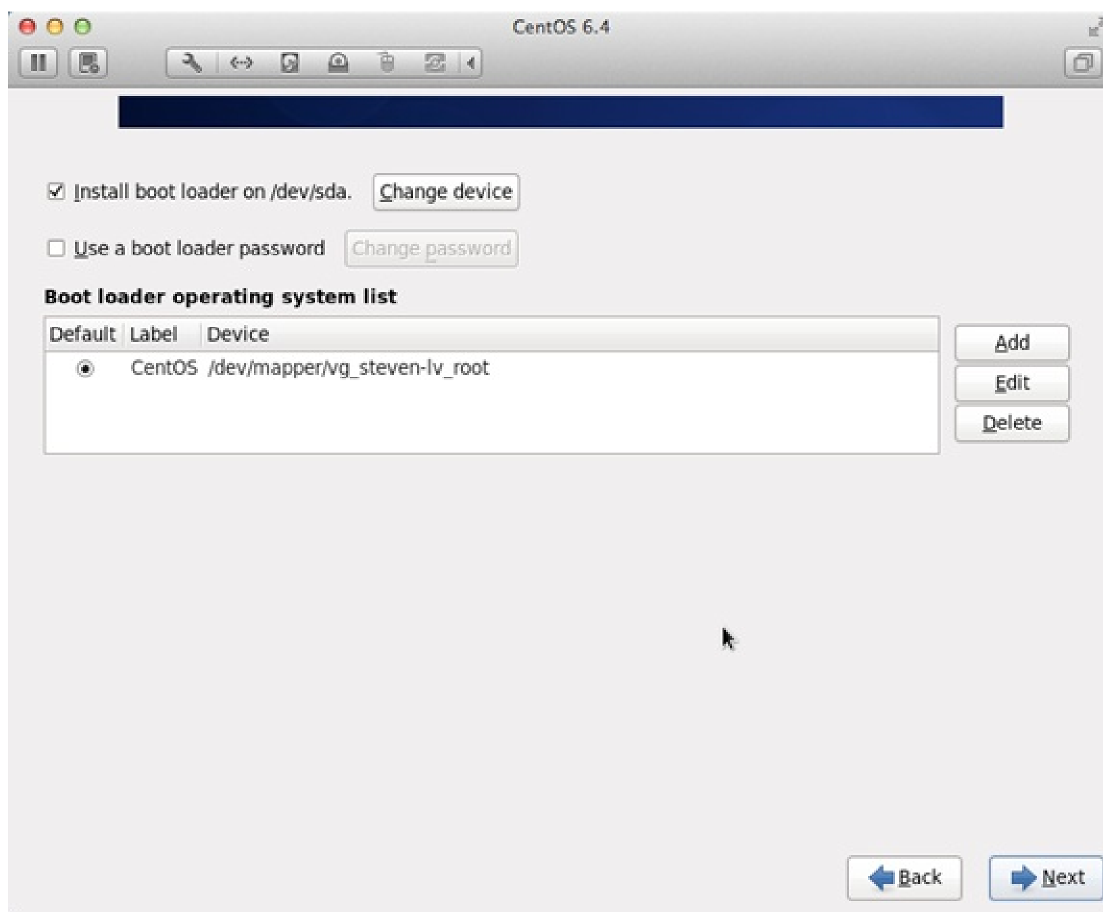
14、询问是否格式化分区



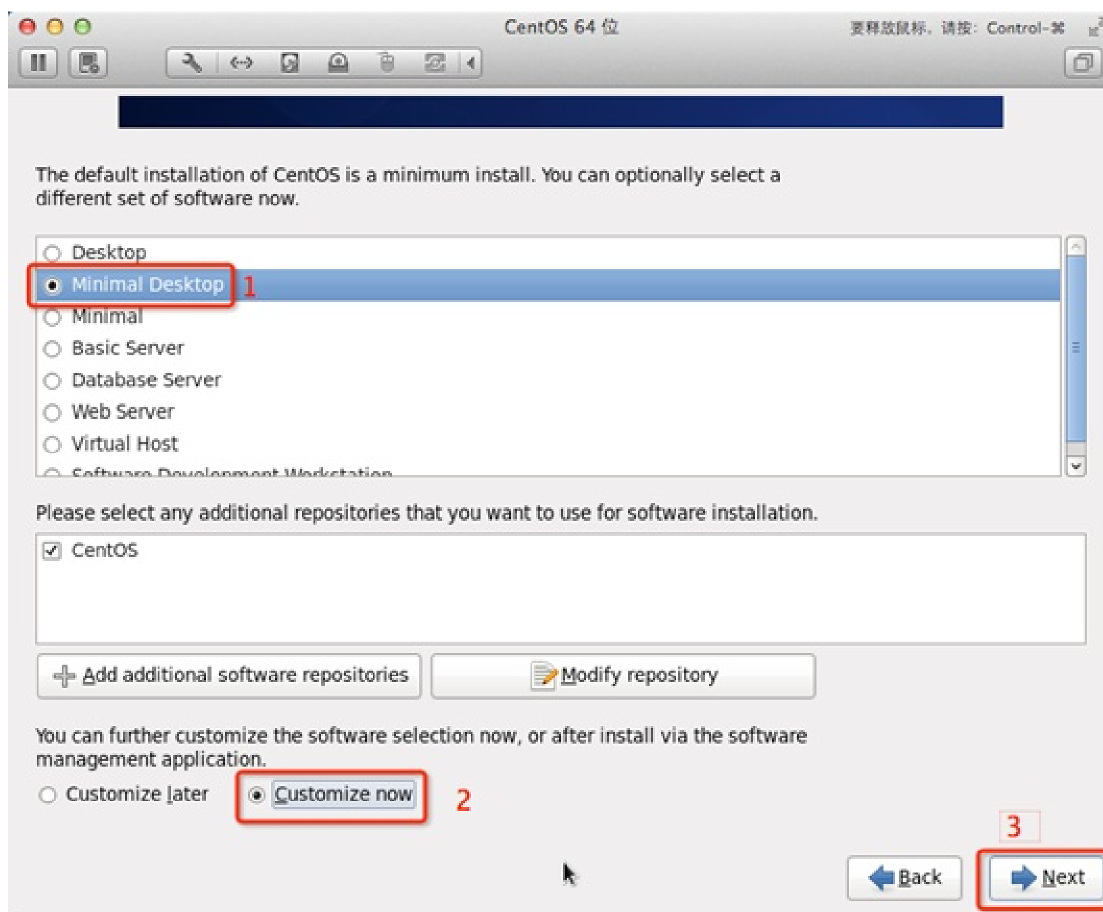
15、将更改写入到硬盘



16、引导程序安装位置



17、最重要的一步，也是本教程最关机的一步，也是其他教程没有提及的一步，按图示顺序点击



18、取消以下内容的所有选项

Applications

Base System

Servers

并对Desktops进行如下设置

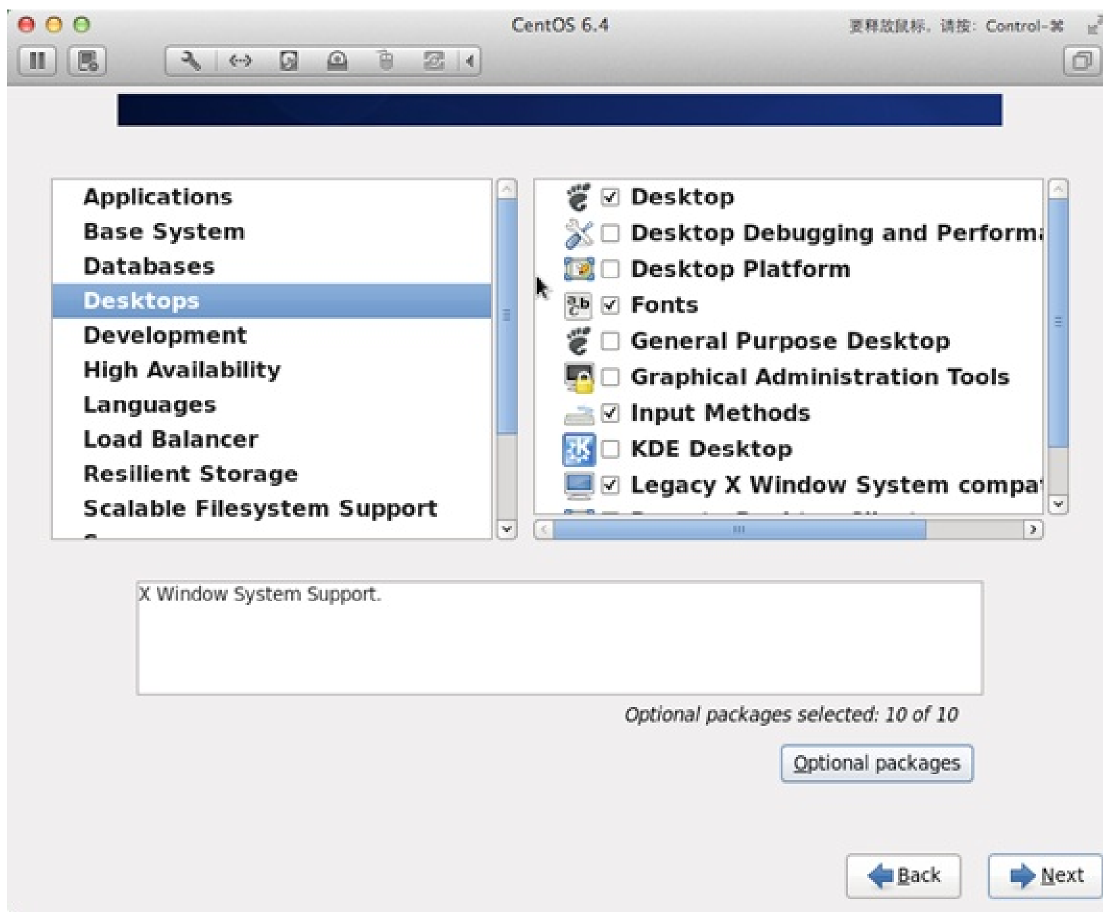
即取消如下选项：

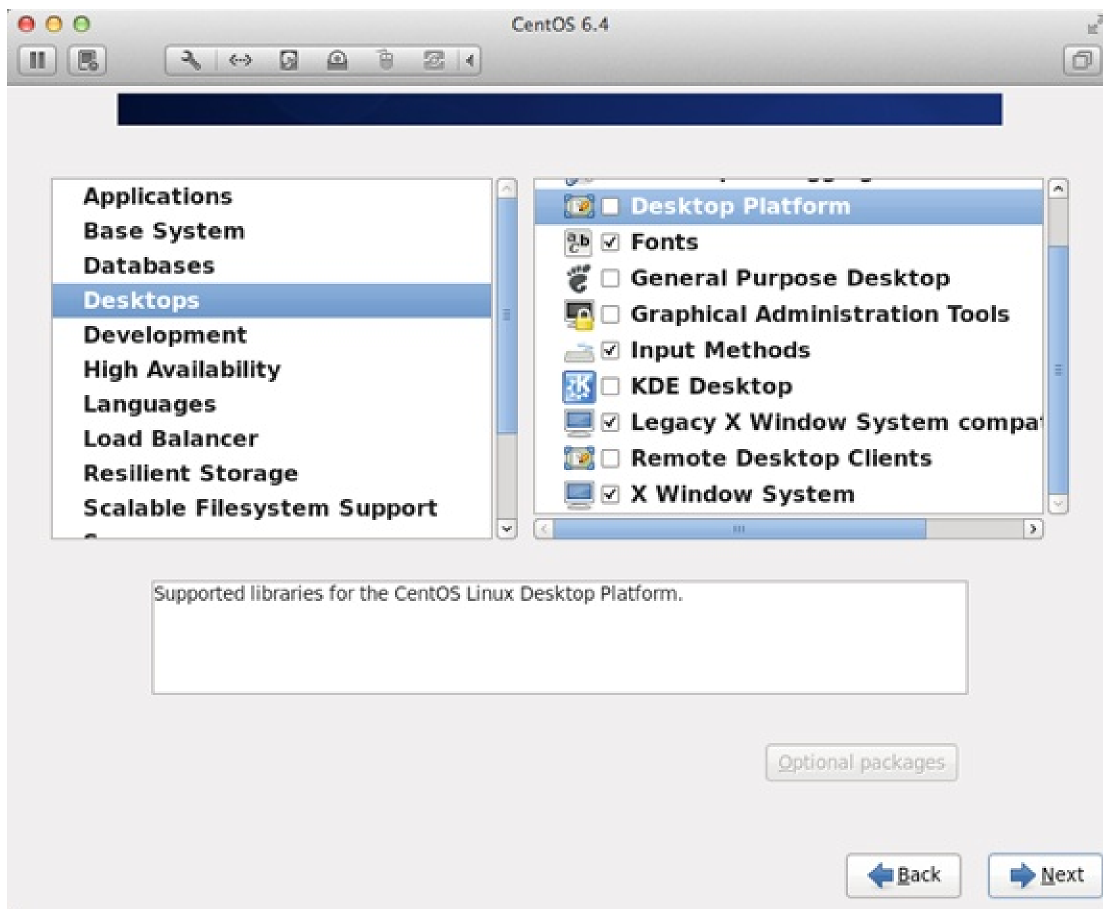
Desktop Debugging and Performance Tools

Desktop Platform

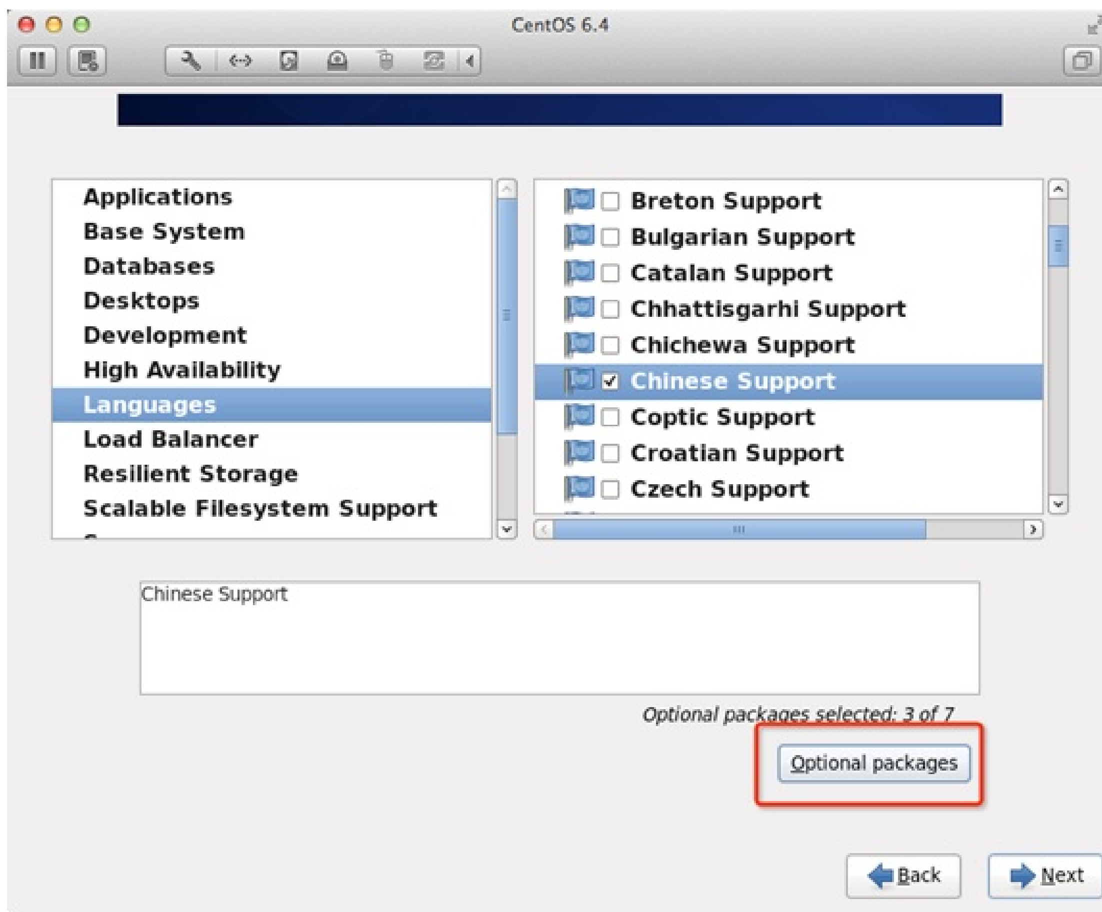
Remote Desktop Clients

Input Methods中仅保留ibus-pinyin-1.3.8-1.el6.x86_64,其他的全部取消**

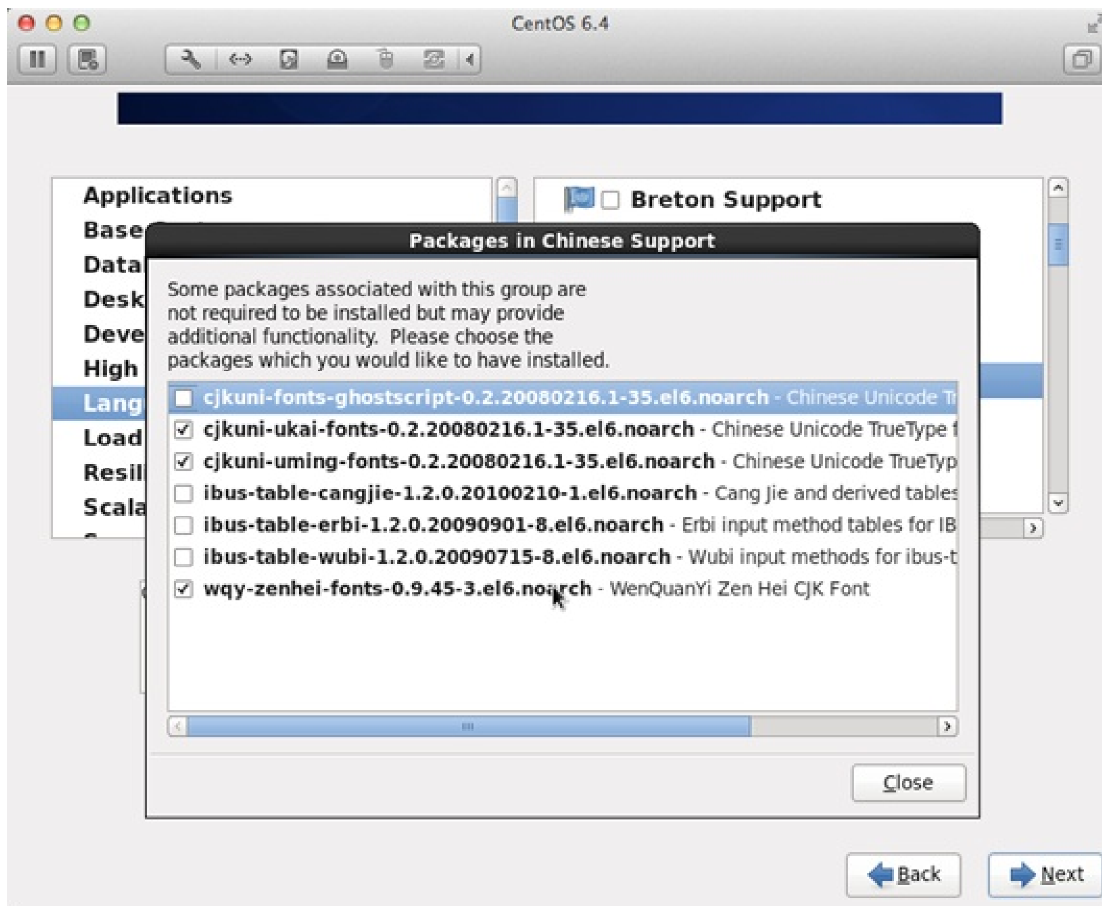




19、选中Languages，并选中右侧的Chinese Support然后点击红色区域



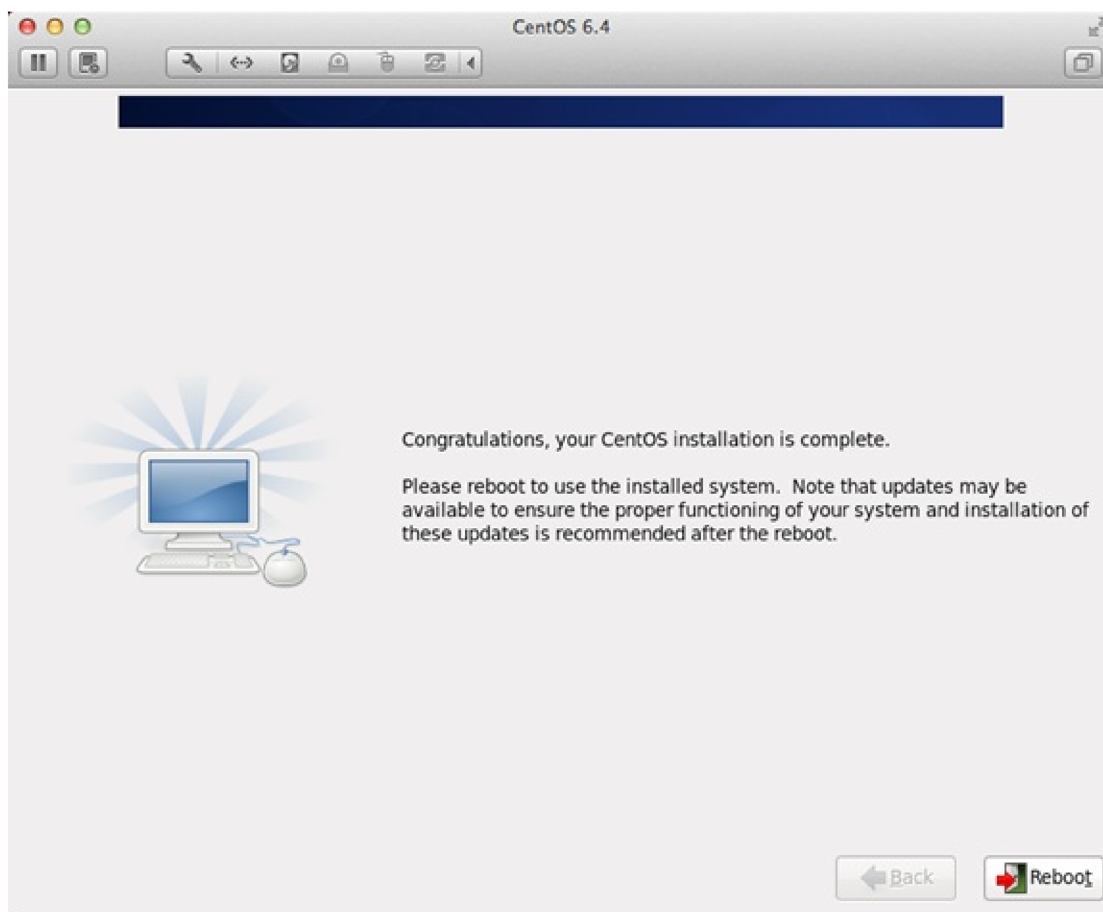
20、调整完成后如下图所示



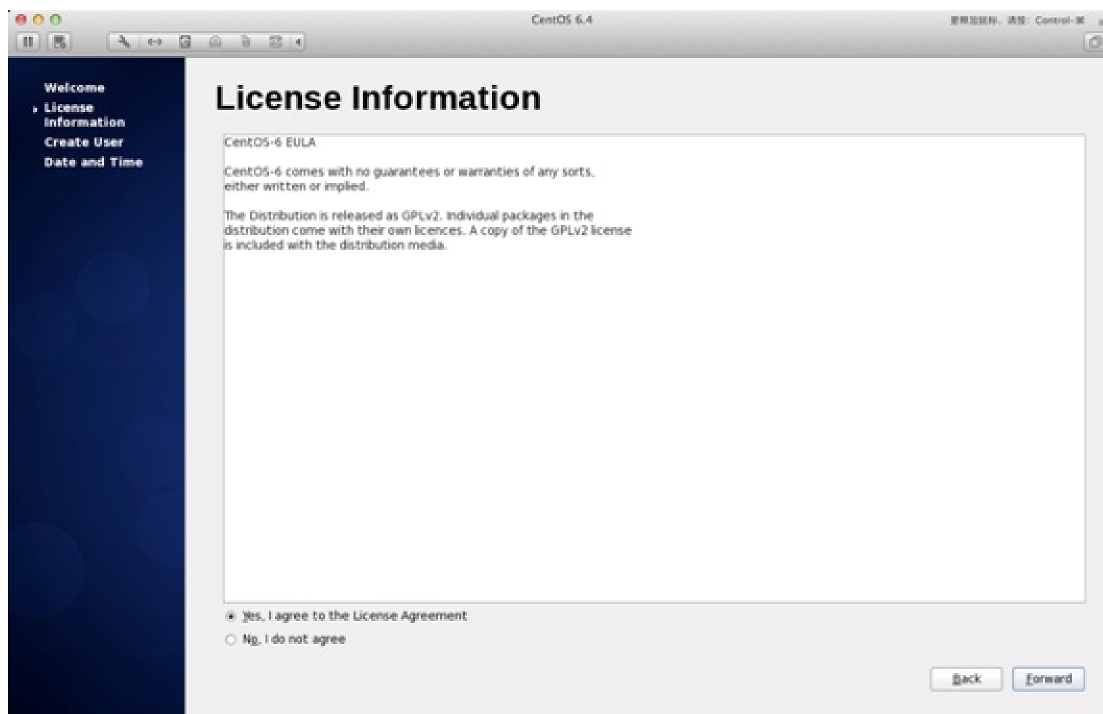
21、至此，一个最精简的桌面环境就设置完成了，



22、安装完成，重启



23、重启之后，的License Information



24、Create User

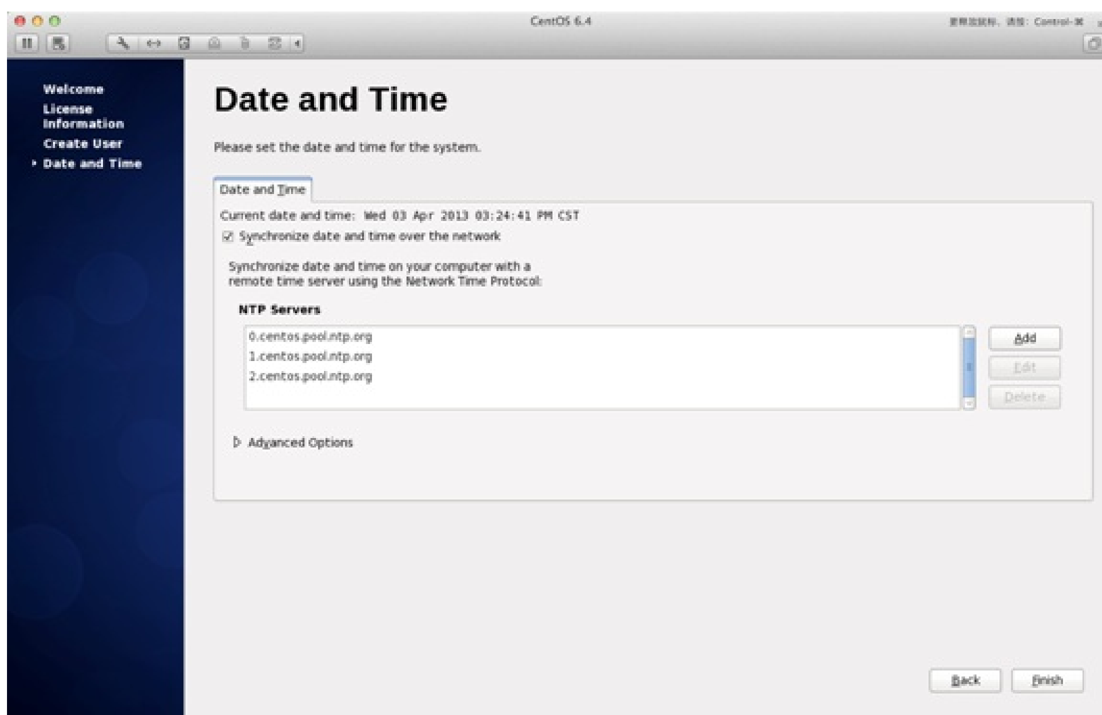
Username：填写您的英文名（不带.姓）

Full Name：填写您的英文名.姓（首字母大写）



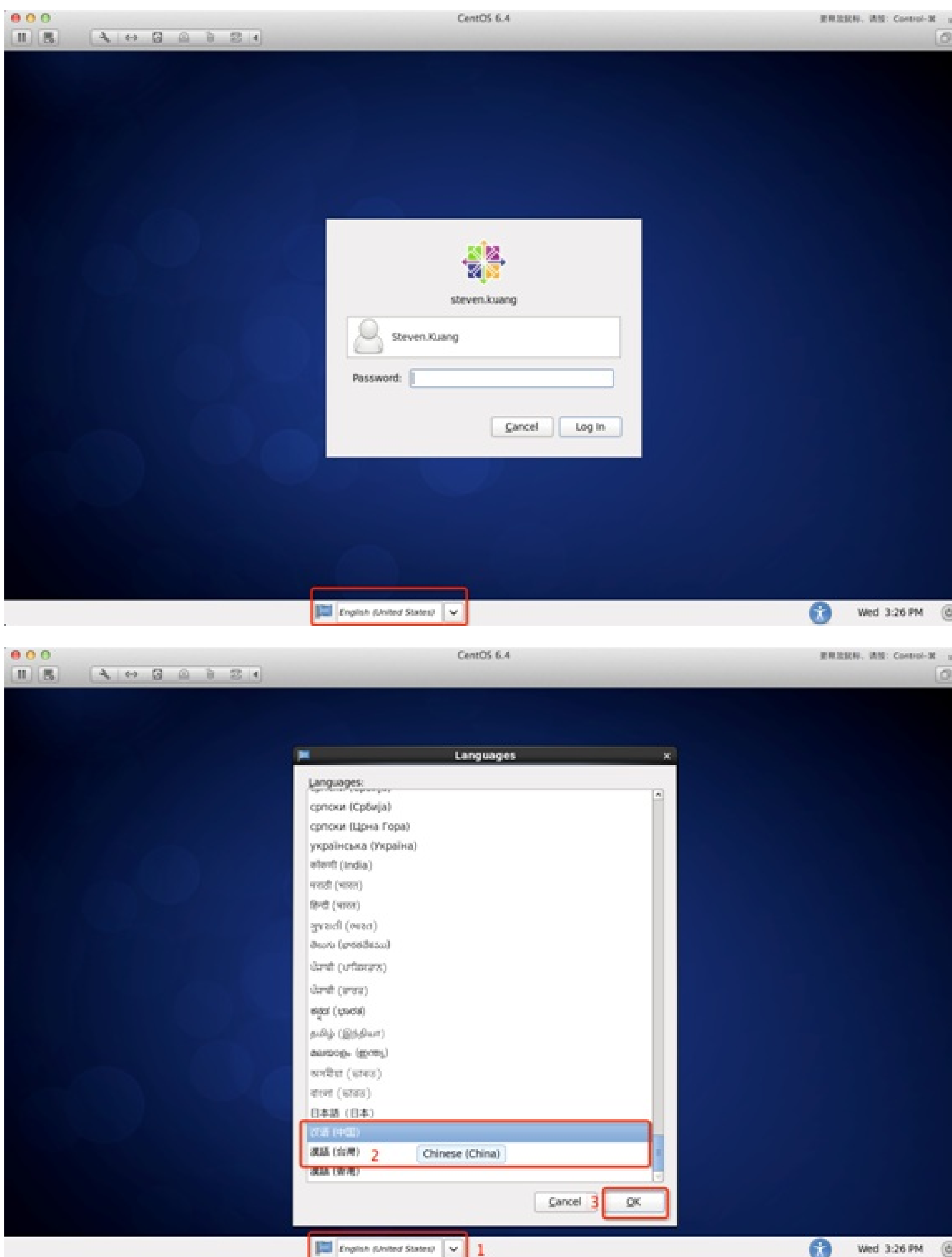
25、"Date and Time" 选中 "Synchronize data and time over the network"

Finsh之后系统将重启



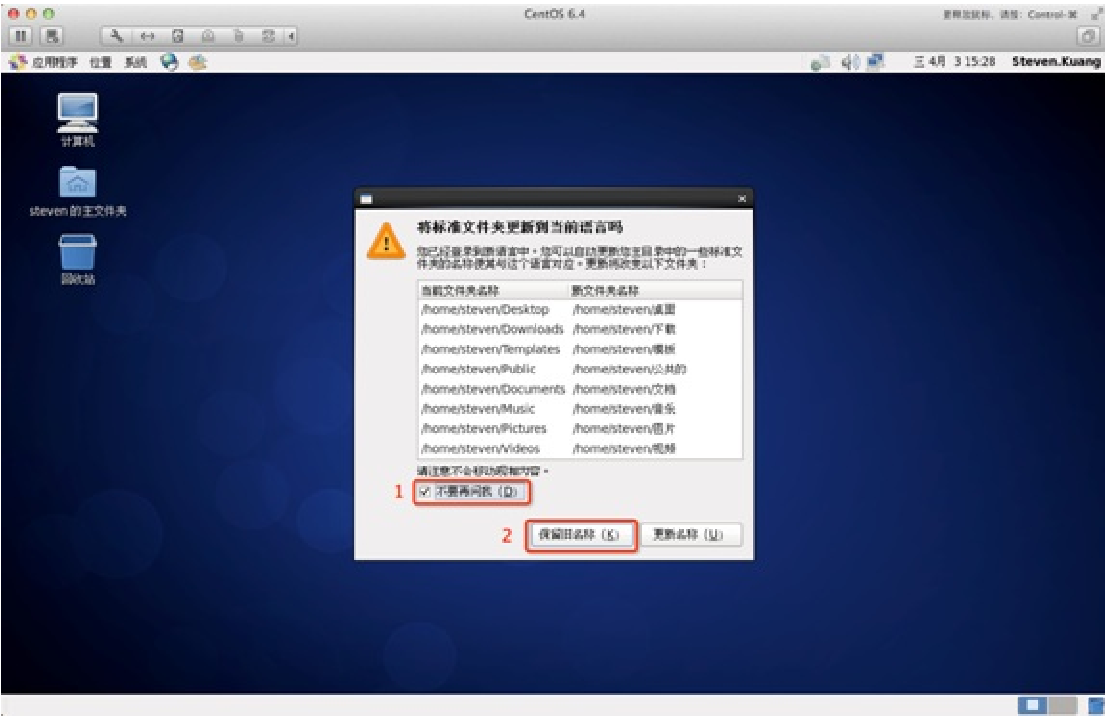
26、第一次登录，登录前不要做任何更改，这个很重要！！！登录之后紧接着退出

第二次登录，选择语言，在红色区域选择下拉小三角，选other，选中"汉语（中国）"



27、登录之后，请一定按照如下顺序点击！

至此，CentOS安装完成，如有其他问题，请随时与我联系！！


CentOS 6.4
计算机
steven 的主文件夹
回收站
Steven.Kuang
将标准文件夹更新到当前语言吗
当前文件夹名称
新文件夹名称
/home/steven/Desktop
/home/steven/桌面
/home/steven/Downloads
/home/steven/下载
/home/steven/Templates
/home/steven/模板
/home/steven/Public
/home/steven/公共的
/home/steven/Documents
/home/steven/文档
/home/steven/Music
/home/steven/音乐
/home/steven/Pictures
/home/steven/图片
/home/steven/Videos
/home/steven/视频
1
不要再问我（D）
2
保留旧名称（K）
更新名称（U）

Linux 系统启动过程

linux启动时我们会看到许多启动信息。

Linux系统的启动过程并不是大家想象中的那么复杂，其过程可以分为5个阶段：

- 内核的引导。
- 运行init。
- 系统初始化。
- 建立终端 。
- 用户登录系统。

内核引导

当计算机打开电源后，首先是BIOS开机自检，按照BIOS中设置的启动设备（通常是硬盘）来启动。

操作系统接管硬件以后，首先读入 /boot 目录下的内核文件。



运行init

init 进程是系统所有进程的起点，你可以把它比拟成系统所有进程的老祖宗，没有这个进程，系统中任何进程都不会启动。

init 程序首先是需要读取配置文件 /etc/inittab。



运行级别

许多程序需要开机启动。它们在Windows叫做"服务"（service），在Linux就叫做"守护进程"（daemon）。

init进程的一大任务，就是去运行这些开机启动的程序。

但是，不同的场合需要启动不同的程序，比如用作服务器时，需要启动Apache，用作桌面就不需要。

Linux允许为不同的场合，分配不同的开机启动程序，这就叫做"运行级别"（runlevel）。也就是说，启动时根据"运行级别"，确定要运行哪些程序。



Linux系统有7个运行级别(runlevel)：

- 运行级别0：系统停机状态，系统默认运行级别不能设为0，否则不能正常启动
- 运行级别1：单用户工作状态，root权限，用于系统维护，禁止远程登陆
- 运行级别2：多用户状态(没有NFS)
- 运行级别3：完全的多用户状态(有NFS)，登陆后进入控制台命令行模式
- 运行级别4：系统未使用，保留
- 运行级别5：X11控制台，登陆后进入图形GUI模式
- 运行级别6：系统正常关闭并重启，默认运行级别不能设为6，否则不能正常启动

系统初始化

在init的配置文件中有这么一行： si::sysinit:/etc/rc.d/rc.sysinit　它调用执行了/etc/rc.d/rc.sysinit，而rc.sysinit是一个bash shell的脚本，它主要是完成一些系统初始化的工作，rc.sysinit是每一个运行级别都要首先运行的重要脚本。

它主要完成的工作有：激活交换分区，检查磁盘，加载硬件模块以及其它一些需要优先执行任务。

```
l5:5:wait:/etc/rc.d/rc 5
```

这一行表示以5为参数运行/etc/rc.d/rc，/etc/rc.d/rc是一个Shell脚本，它接受5作为参数，去执行/etc/rc.d/rc5.d/目录下的所有的rc启动脚本，/etc/rc.d/rc5.d/目录中的这些启动脚本实际上都是一些连接文件，而不是真正的rc启动脚本，真正的rc启动脚本实际上都是放在/etc/rc.d/init.d/目录下。

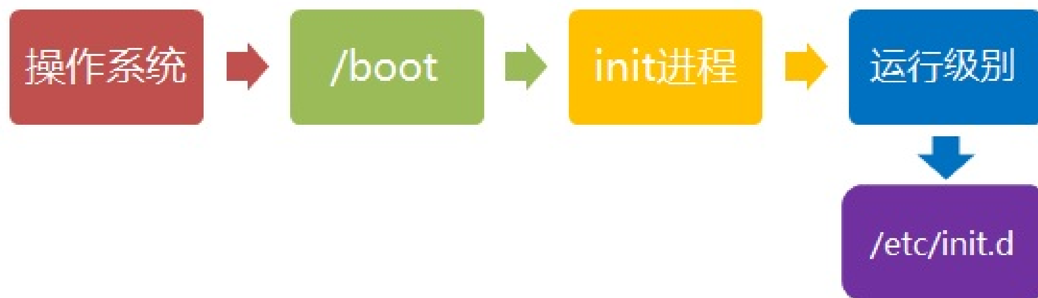
而这些rc启动脚本有着类似的用法，它们一般能接受start、stop、restart、status等参数。

/etc/rc.d/rc5.d/中的rc启动脚本通常是K或S开头的连接文件，对于以以S开头的启动脚本，将以start参数来运行。

而如果发现存在相应的脚本也存在K打头的连接，而且已经处于运行态了(以/var/lock/subsys/下的文件作为标志)，则将首先以stop为参数停止这些已经启动了的守护进程，然后再重新运行。

这样做是为了保证是当init改变运行级别时，所有相关的守护进程都将重启。

至于在每个运行级中将运行哪些守护进程，用户可以通过chkconfig或setup中的"System Services"来自行设定。



建立终端

rc执行完毕后，返回init。这时基本系统环境已经设置好了，各种守护进程也已经启动了。

init接下来会打开6个终端，以便用户登录系统。在inittab中的以下6行就是定义了6个终端：

```
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

从上面可以看出在2、3、4、5的运行级别中都将以respawn方式运行mingetty程序，mingetty程序能打开终端、设置模式。

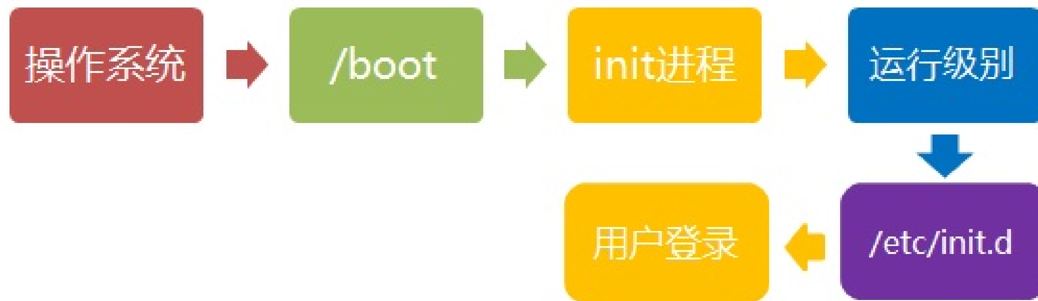
同时它会显示一个文本登录界面，这个界面就是我们经常看到的登录界面，在这个登录界面中会提示用户输入用户名，而用户输入的用户将作为参数传给login程序来验证用户的身份。

用户登录系统

一般来说，用户的登录方式有三种：

- （1）命令行登录

- （2）ssh登录
- （3）图形界面登录



对于运行级别为5的图形方式用户来说，他们的登录是通过一个图形化的登录界面。登录成功后可以直接进入KDE、Gnome等窗口管理器。

而本文主要讲的还是文本方式登录的情况：当我们看到mingetty的登录界面时，我们就可以输入用户名和密码来登录系统了。

Linux的账号验证程序是login，login会接收mingetty传来的用户名作为用户名参数。

然后login会对用户名进行分析：如果用户名不是root，且存在/etc/nologin文件，login将输出nologin文件的内容，然后退出。

这通常用来系统维护时防止非root用户登录。只有/etc/securetty中登记了的终端才允许root用户登录，如果不存在这个文件，则root可以在任何终端上登录。

/etc/usertty文件用于对用户作出附加访问限制，如果不存在这个文件，则没有其他限制。

图形模式与文字模式的切换方式

Linux预设提供了六个命令窗口终端机让我们来登录。

默认我们登录的就是第一个窗口，也就是tty1，这个六个窗口分别为tty1,tty2 … tty6，你可以按下Ctrl + Alt + F1 ~ F6 来切换它们。

如果你安装了图形界面，默认情况下是进入图形界面的，此时你就可以按Ctrl + Alt + F1 ~ F6来进入其中一个命令窗口界面。

当你进入命令窗口界面后再返回图形界面只要按下Ctrl + Alt + F7 就回来了。

如果你用的vmware 虚拟机，命令窗口切换的快捷键为 Alt + Space + F1~F6. 如果你在图形界面下请按Alt + Shift + Ctrl + F1~F6 切换至命令窗口。



Linux 关机

在linux领域内大多用在服务器上，很少遇到关机的操作。毕竟服务器上跑一个服务是永无止境的，除非特殊情况下，不得已才会关机。

正确的关机流程为：sysnc > shutdown > reboot > halt

关机指令为：shutdown ， 你可以man shutdown 来看一下帮助文档。

例如你可以运行如下命令关机：

```
sync 将数据由内存同步到硬盘中。

shutdown 关机指令，你可以man shutdown 来看一下帮助文档。例如你可以运行如下命令关机：

shutdown –h 10 ‘This server will shutdown after 10 mins’ 这个命令告诉大家，计算机将在10分钟后关

Shutdown –h now 立马关机

Shutdown –h 20:25 系统会在今天20:25关机

Shutdown –h +10 十分钟后关机

Shutdown –r now 系统立马重启

Shutdown –r +10 系统十分钟后重启

reboot 就是重启，等同于 shutdown –r now

halt 关闭系统，等同于shutdown –h now 和 poweroff
```

最后总结一下，不管是重启系统还是关闭系统，首先要运行sync命令，把内存中的数据写到磁盘中。

关机的命令有 shutdown –h now halt poweroff 和 init 0 , 重启系统的命令有 shutdown –r now reboot init 6.

Linux 系统目录结构

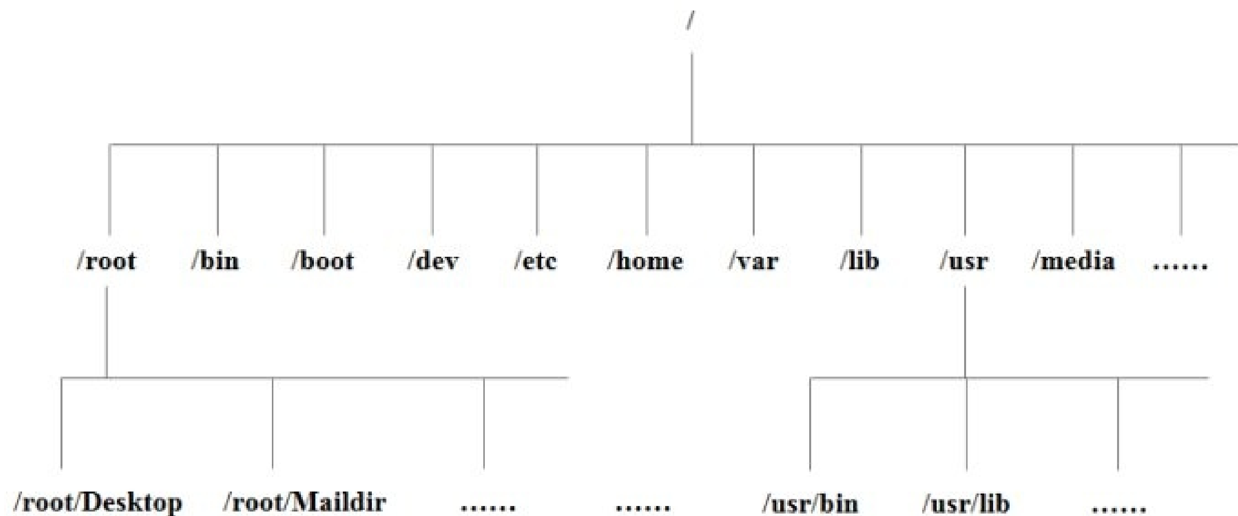
登录系统后，在当前命令窗口下输入命令：

```
ls /
```

你会看到如下图所示:

```
[root@localhost ~]# ls /
bin   dev  home  lost+found  mnt  proc  sbin     srv  tmp  var
boot  etc  lib   media       opt  root  selinux  sys  usr
```

树状目录结构：



以下是对这些目录的解释：

- **/bin**：
  bin是Binary的缩写, 这个目录存放着最经常使用的命令。

- **/boot**：
  这里存放的是启动Linux时使用的一些核心文件，包括一些连接文件以及镜像文件。

- **/dev** ：
  dev是Device(设备)的缩写, 该目录下存放的是Linux的外部设备，在Linux中访问设备的方式和访问文件的方式是相同的。

- **/etc**：
  这个目录用来存放所有的系统管理所需要的配置文件和子目录。

- **/home**：
  用户的主目录，在Linux中，每个用户都有一个自己的目录，一般该目录名是以用户的账号命名的。

- **/lib**：
  这个目录里存放着系统最基本的动态连接共享库，其作用类似于Windows里的DLL文件。几乎所有的应用程序都需要用到这些共享库。

- **/lost+found**：
  这个目录一般情况下是空的，当系统非法关机后，这里就存放了一些文件。

- /media linux系统会自动识别一些设备，例如U盘、光驱等等，当识别后，linux会把识别的设备挂载到这个目录下。

- **/mnt**：
  系统提供该目录是为了让用户临时挂载别的文件系统的，我们可以将光驱挂载在/mnt/上，然后进入该目录就可以查看光驱里的内容了。

- **/opt**：
  这是给主机额外安装软件所摆放的目录。比如你安装一个ORACLE数据库则就可以放到这个目录下。默认是空的。

- **/proc**：
  这个目录是一个虚拟的目录，它是系统内存的映射，我们可以通过直接访问这个目录来获取系统信息。
  这个目录的内容不在硬盘上而是在内存里，我们也可以直接修改里面的某些文件，比如可以通过下面的命令来屏蔽主机的ping命令，使别人无法ping你的机器：

  ```
  echo 1 &gt; /proc/sys/net/ipv4/icmp_echo_ignore_all
  ```

- **/root**：
  该目录为系统管理员，也称作超级权限者的用户主目录。

- **/sbin**：
  s就是Super User的意思，这里存放的是系统管理员使用的系统管理程序。

- **/selinux**：
  这个目录是Redhat/CentOS所特有的目录，Selinux是一个安全机制，类似于windows的防火墙，但是这套机制比较复杂，这个目录就是存放selinux相关的文件的。

- **/srv**：
  该目录存放一些服务启动之后需要提取的数据。

- **/sys**：
  这是linux2.6内核的一个很大的变化。该目录下安装了2.6内核中新出现的一个文件系统 sysfs 。

  sysfs文件系统集成了下面3种文件系统的信息：针对进程信息的proc文件系统、针对设备的devfs文件系统以及针对伪终端的devpts文件系统。

该文件系统是内核设备树的一个直观反映。

当一个内核对象被创建的时候，对应的文件和目录也在内核对象子系统种被创建。

- **/tmp**：
  这个目录是用来存放一些临时文件的。

- **/usr**：
  这是一个非常重要的目录，用户的很多应用程序和文件都放在这个目录下，类似与windows下的program files目录。

- **/usr/bin**：
  系统用户使用的应用程序。

- **/usr/sbin**：
  超级用户使用的比较高级的管理程序和系统守护程序。

- **/usr/src**：内核源代码默认的放置目录。

- **/var**：
  这个目录中存放着在不断扩充着的东西，我们习惯将那些经常被修改的目录放在这个目录下。包括各种日志文件。

在linux系统中，有几个目录是比较重要的，平时需要注意不要误删除或者随意更改内部文件。

/etc： 上边也提到了，这个是系统中的配置文件，如果你更改了该目录下的某个文件可能会导致系统不能启动。

/bin, /sbin, /usr/bin, /usr/sbin: 这是系统预设的执行文件的放置目录，比如 ls 就是在/bin/ls 目录下的。

值得提出的是，/bin, /usr/bin 是给系统用户使用的指令（除root外的通用户），而/sbin, /usr/sbin 则是给root使用的指令。

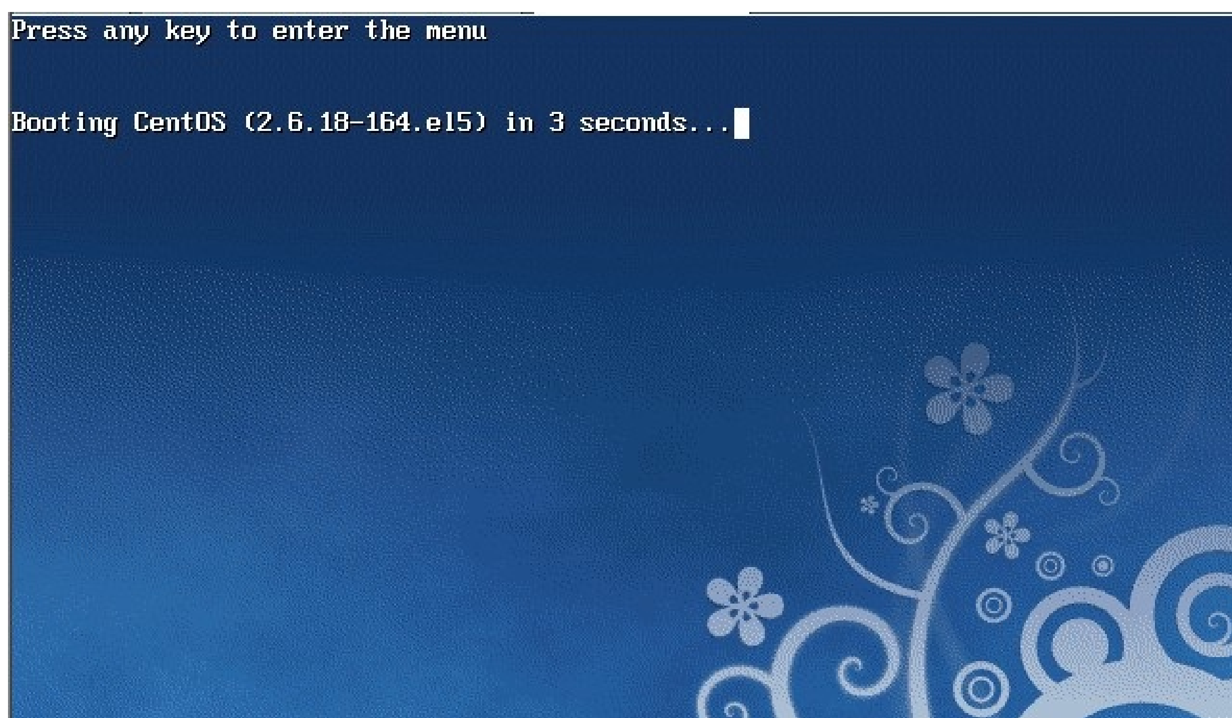
/var： 这是一个非常重要的目录，系统上跑了很多程序，那么每个程序都会有相应的日志产生，而这些日志就被记录到这个目录下，具体在/var/log 目录下，另外mail的预设放置也是在这里。

Linux 忘记密码解决方法

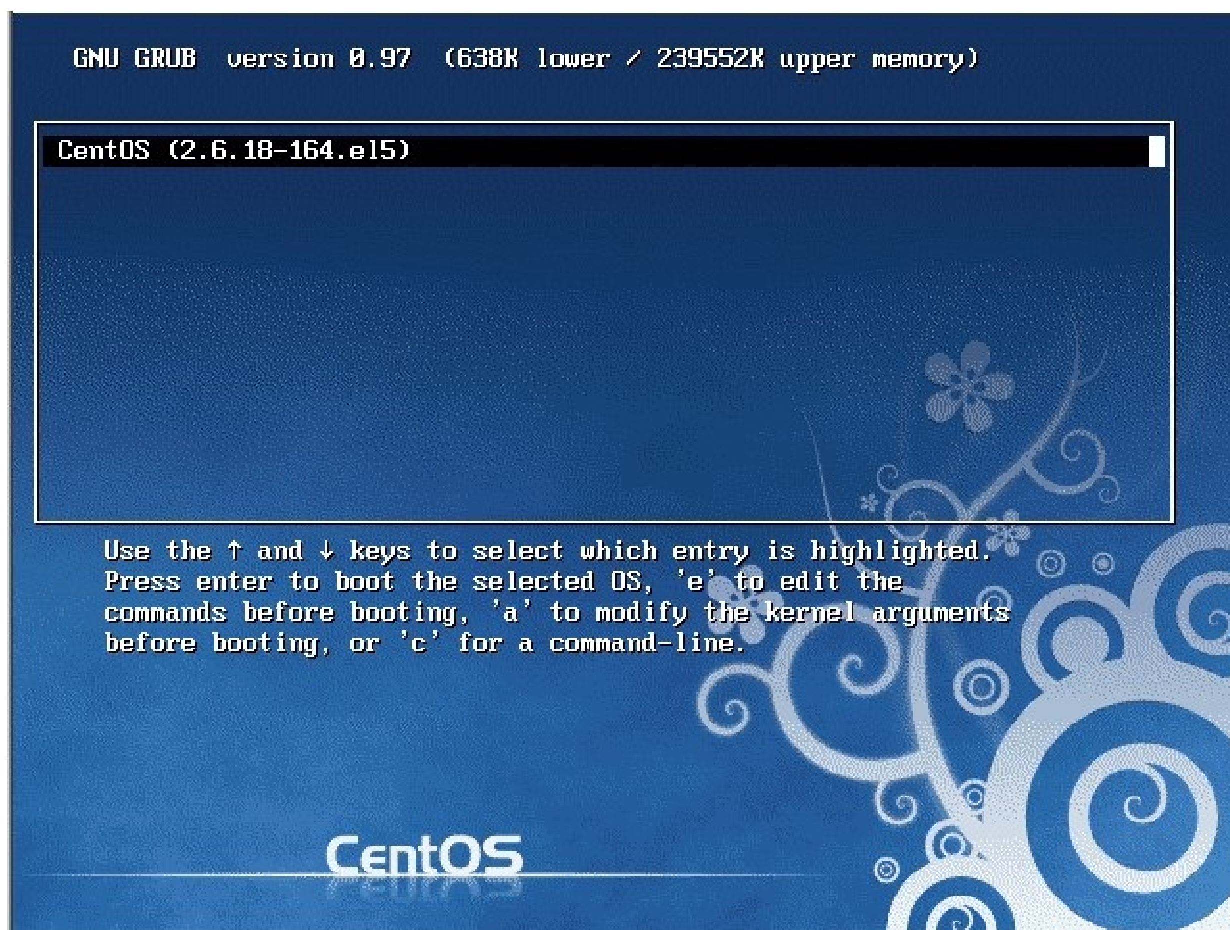
很多朋友经常会忘记Linux系统的root密码，linux系统忘记root密码的情况该怎么办呢？重新安装系统吗？当然不用！进入单用户模式更改一下root密码即可。

步骤如下：

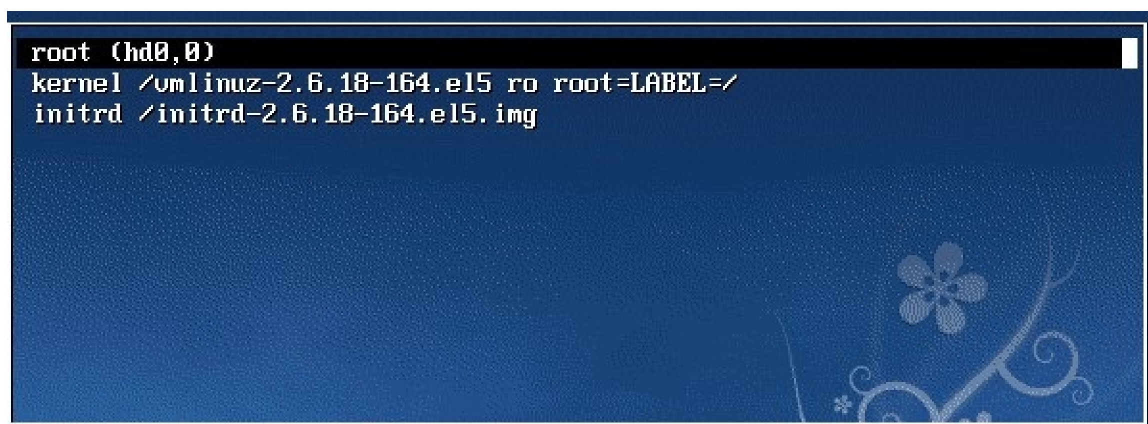
重启linux系统



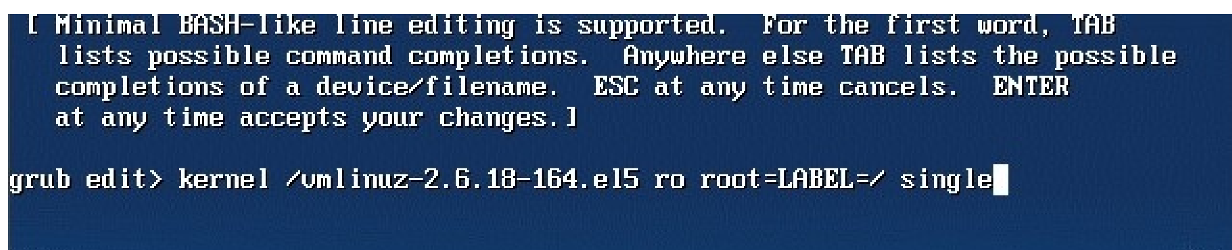
3 秒之内要按一下回车，出现如下界面



然后输入e



在 第二行最后边输入 single，有一个空格。具体方法为按向下尖头移动到第二行，按"e"进入编辑模式



在后边加上single 回车

```
root (hd0,0)
kernel /vmlinuz-2.6.18-164.el5 ro root=LABEL=/ single
initrd /initrd-2.6.18-164.el5.img
```

最后按"b"启动，启动后就进入了单用户模式了

```
no fstab.sys, mounting internal defaults
Switching to new root and running init.
unmounting old /dev
unmounting old /proc
unmounting old /sys
type=1404 audit(1303914022.636:2): enforcing=1 old_enforcing=0 auid=4294967295 s
es=4294967295
type=1403 audit(1303914023.222:3): policy loaded auid=4294967295 ses=4294967295
INIT: version 2.86 booting
                Welcome to  CentOS release 5.4 (Final)
                Press 'I' to enter interactive startup.
Setting clock  (utc): Wed Apr 27 22:20:50 CST 2011         [  OK  ]
Starting udev:                                             [  OK  ]
Loading default keymap (us):                               [  OK  ]
Setting hostname localhost.localdomain:                    [  OK  ]
No devices found
Setting up Logical Volume Management:                      [  OK  ]
Checking filesystems
/: clean, 118508/1969568 files, 713597/1967962 blocks
/boot: clean, 35/26104 files, 14714/104388 blocks
                                                           [  OK  ]
Remounting root filesystem in read-write mode:             [  OK  ]
Mounting local filesystems:                                [  OK  ]
Enabling /etc/fstab swaps:                                 [  OK  ]
sh-3.2# _
```

此时已经进入到单用户模式了，你可以更改root密码了。更密码的命令为 passwd

```
sh-3.2# passwd
Changing password for user root.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

【使用系统安装光盘的救援模式】

救援模式即rescue ，这个模式主要是应用于，系统无法进入的情况。如，grub损坏或者某一个配置文件修改出错。如何使用rescue模式呢？

光盘启动，按F5 进入rescue模式

```
 -  To install or upgrade in graphical mode, press the <ENTER> key.

 -  To install or upgrade in text mode, type: linux text <ENTER>.

 -  Use the function keys listed below for more information.

[F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]
boot: _
```

输入linux rescue 回车

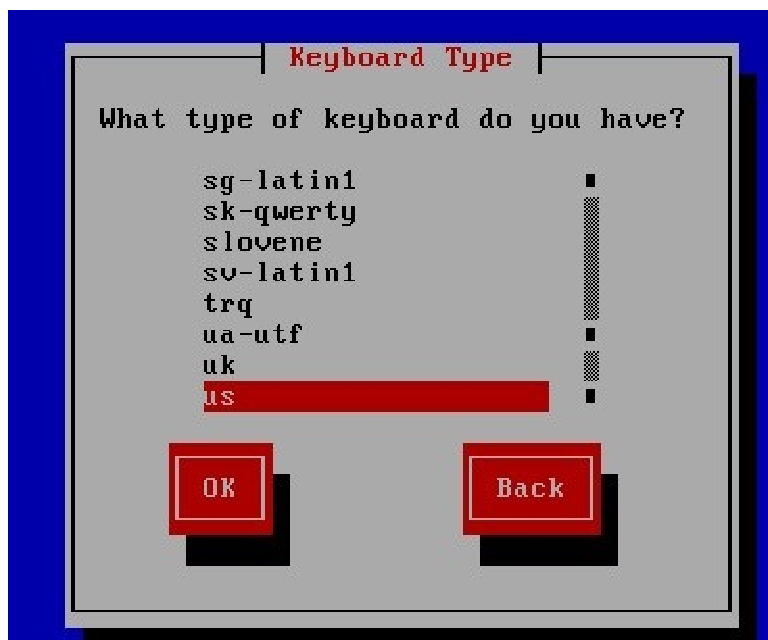
```
[F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]

boot: linux rescue_
```

选择语言，笔者建议你选择英语

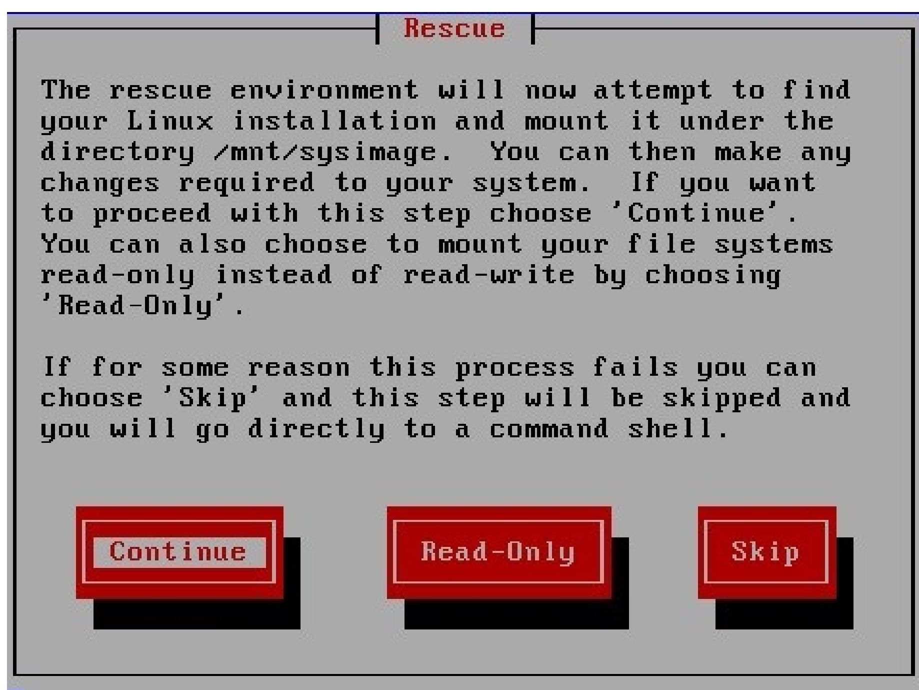


选择us 键盘





这里问你是否启动网络，有时候可能会联网调试。我们选no

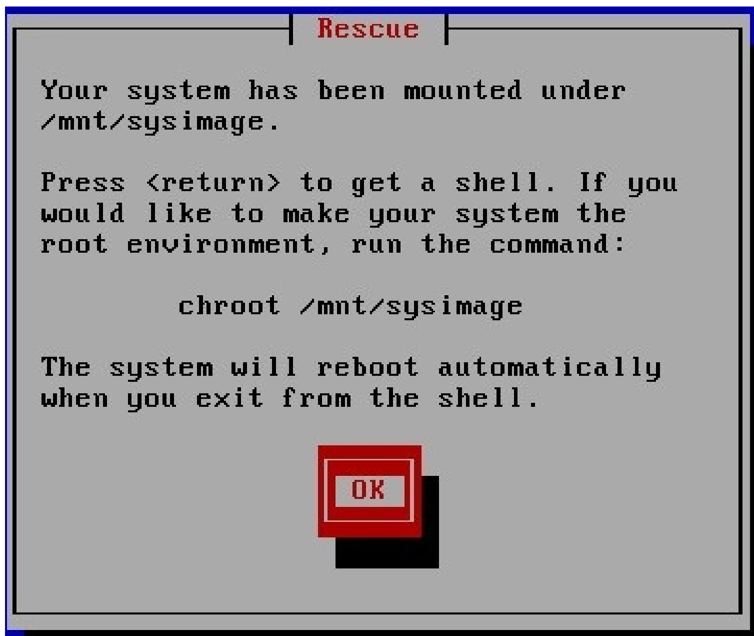


这里告诉我们，接下来会把系统挂载在/mnt/sysimage 中。

其中有三个选项:

- Continue 就是挂载后继续下一步。
- Read-Only 挂载成只读，这样更安全，有时文件系统损坏时，只读模式会防止文件系统近一步损坏。
- Skip就是不挂载，进入一个命令窗口模式。

这里我们选择Continue。



至此，系统已经挂载到了/mnt/sysimage中。接下来回车，输入chroot /mnt/sysimage 进入管理员环境。

```
Your system is mounted under the /mnt/sysimage directory.
When finished please exit from the shell and your system will reboot.

sh-3.2# chroot /mnt/sysimage/
sh-3.2# _
```

提示： 其实也可以到rescue模式下更改root的密码的。这个rescue模式和windows PE系统很相近。

当运行了chroot /mnt/sysimage/ 后，再ls 看到目录结构和原来系统中的目录结构是一样的。

没错！现在的环境和原来系统的环境是一模一样的。你可以输入exit 或者按Ctrl + D退出这个环境。然后你再ls 看一下

```
sh-3.2# ls
bin  etc   lib  modules  proc  sbin     sys  usr
dev  init  mnt  oldtmp   root  selinux  tmp  var
sh-3.2# ls /mnt/
runtime  source  sysimage
sh-3.2# _
```

这个目录其实就是rescue模式下的目录结构，而我们的系统文件全部在 /mnt/sysimage目录下。

Linux 远程登录

Linux一般作为服务器使用，而服务器一般放在机房，你不可能在机房操作你的Linux服务器。

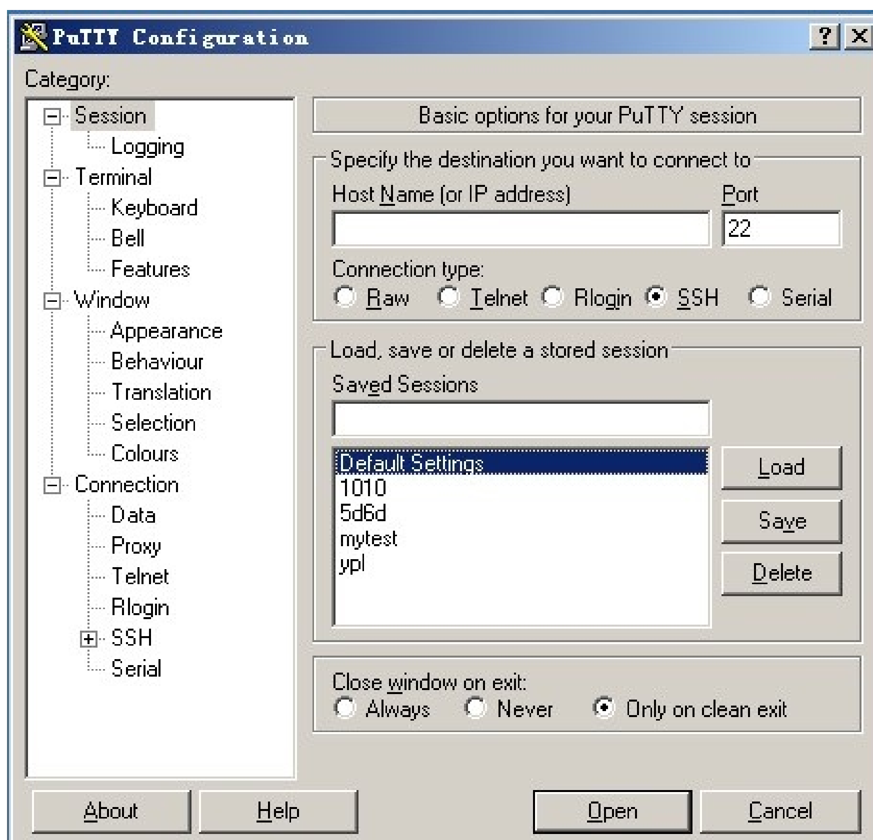
这事我们就需要远程登录到Linux服务器来管理维护系统。

Linux系统中是通过ssh服务实现的远程登录功能，默认ssh服务端口号为 22。

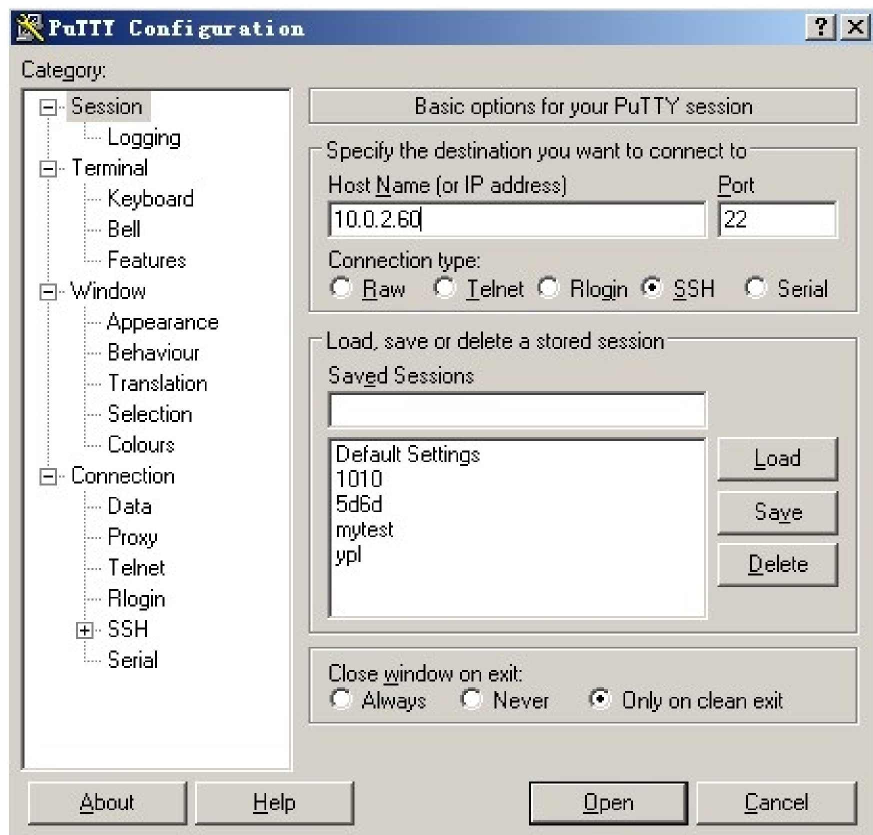
Window系统上 Linux 远程登录客户端有SecureCRT, Putty, SSH Secure Shell等，本文以Putty为例来登录远程服务器。

putty下载地址：http://www.putty.org/

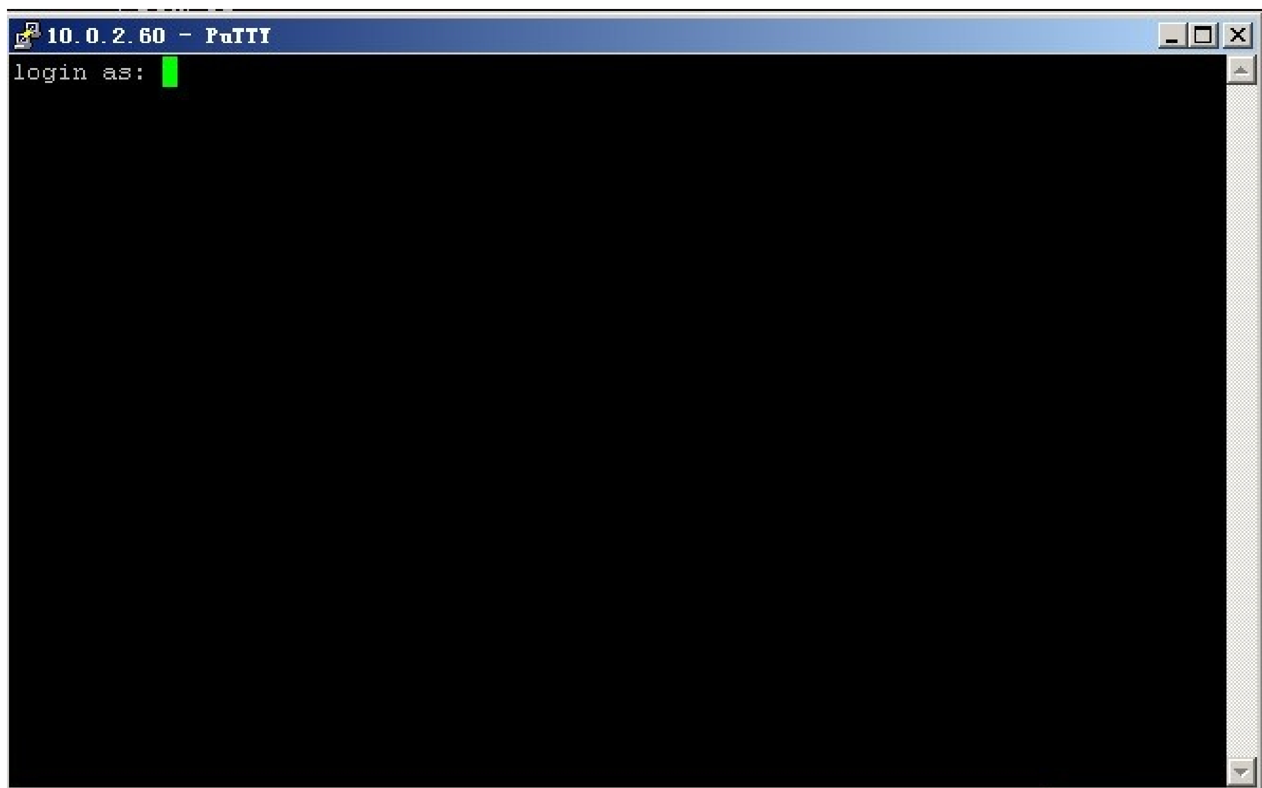
如果你下载了putty，请双击putty.exe 然后弹出如下的窗口。



在Host Name(or IP address) 下面的框中输入你要登录的远程服务器IP(可以通过ifconfig命令查看服务器ip)，然后回车。



此时，提示我们输入要登录的用户名。



输入root 然后回车，再输入密码，就能登录到远程的linux系统了。

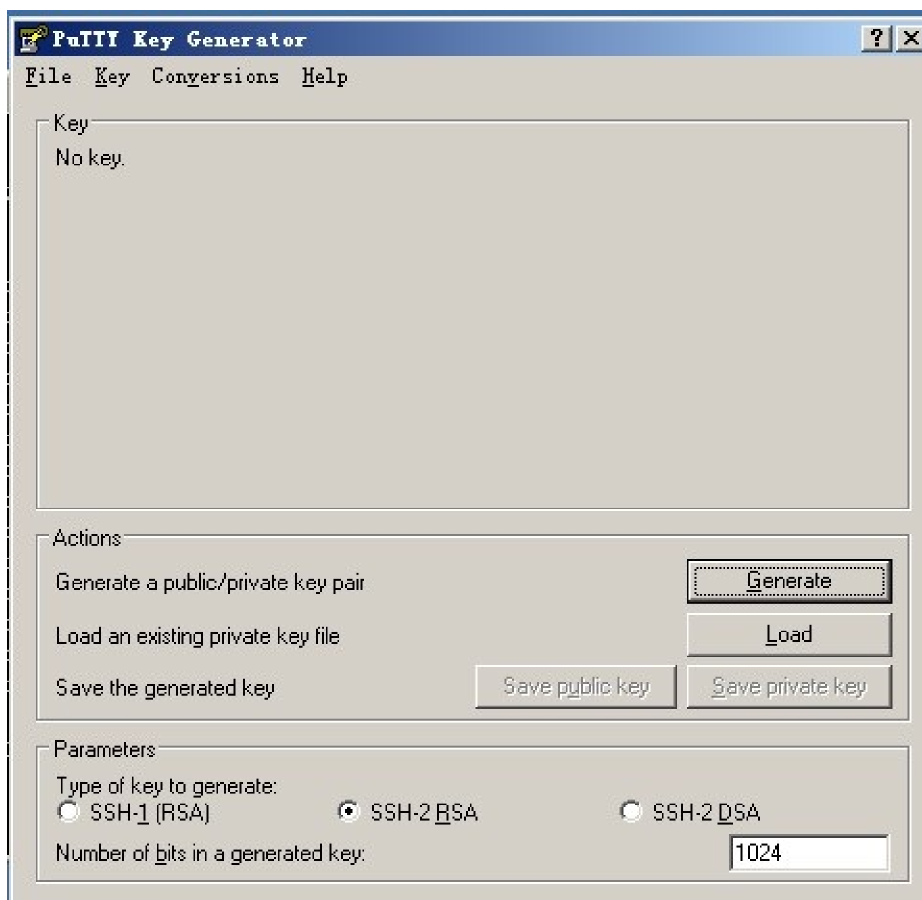
```
root@localhost:~
login as: root
root@10.0.2.60's password:
Last login: Fri Apr 29 14:10:52 2011
[root@localhost ~]#
```

使用密钥认证机制远程登录linux

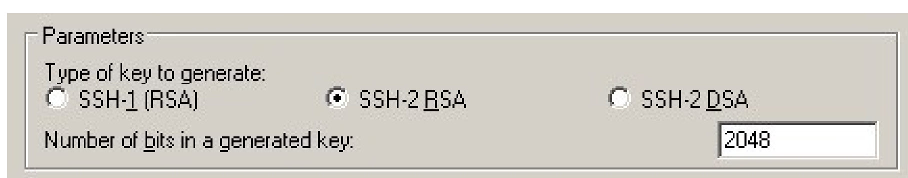
SSH 为 Secure Shell 的缩写，由 IETF 的网络工作小组（Network Working Group）所制定。

SSH 为建立在应用层和传输层基础上的安全协议。

首先使用工具 PUTTYGEN.EXE 生成密钥对。打开工具PUTTYGEN.EXE后如下图所示：



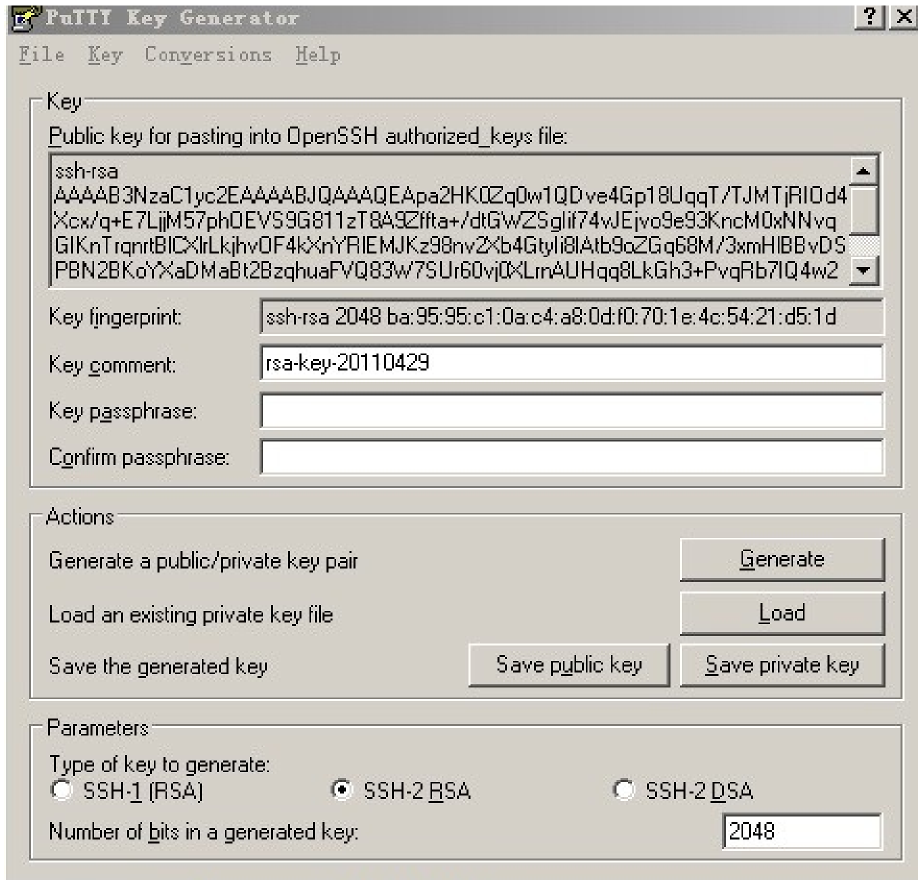
该工具可以生成三种格式的key ：SSH-1(RSA) SSH-2(RSA) SSH-2(DSA)， 我们采用默认的格式即SSH-2(RSA)。Number of bits in a generated key 这个是指生成的key的大小， 这个数值越大， 生成的key就越复杂， 安全性就越高。这里我们写2048.



然后单击Generate 开始生成密钥对：



注意的是，在这个过程中鼠标要来回的动，否则这个进度条是不会动的。



到这里，密钥对已经生成了。你可以给你的密钥输入一个密码，（在Key Passphrase那里）也可以留空。然后点 Save public key 保存公钥，点 Save private Key 保存私钥。笔者建议你放到一个比较安全的地方，一来防止别人偷窥，二来防止误删除。接下来就该到远程linux主机上设置了。

1）创建目录 /root/.ssh 并设置权限

[root@localhost ~]# mkdir /root/.ssh mkdir 命令用来创建目录，以后会详细介绍，暂时只了解即可。

[root@localhost ~]# chmod 700 /root/.ssh chmod 命令是用来修改文件属性权限的，以后会详细介绍。

2）创建文件 / root/.ssh/authorized_keys

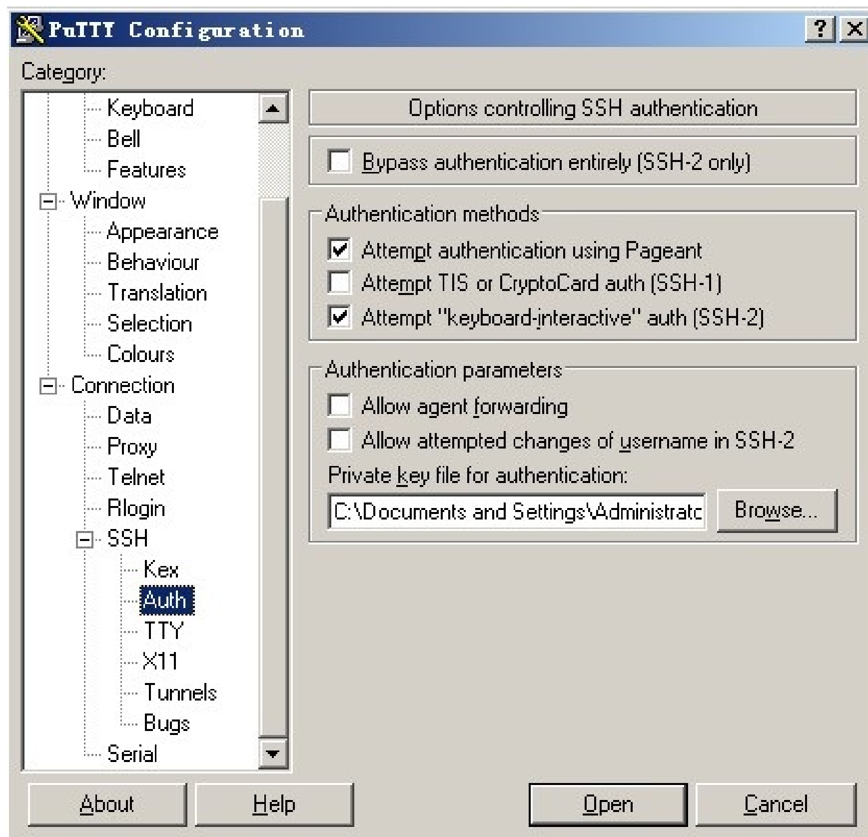
[root@localhost ~]# vim /root/.ssh/authorized_keys vim 命令是编辑一个文本文件的命令，同样在后续章节详细介绍。

3）打开刚才生成的public key 文件，建议使用写字板打开，这样看着舒服一些，复制从AAAA开头至 "---- END SSH2 PUBLIC KEY ----" 该行上的所有内容，粘贴到/root/.ssh/authorized_keys 文件中，要保证所有字符在一行。（可以先把复制的内容拷贝至记事本，然后编辑成一行载粘贴到该文件中）。

在这里要简单介绍一下，如何粘贴，用vim打开那个文件后，该文件不存在，所以vim会自动创建。按一下字母"i"然后同时按shift + Insert 进行粘贴（或者单击鼠标邮件即可），前提是已经复制到剪切板中了。粘贴好后，然后把光标移动到该行最前面输入ssh-ras， 然后按空格。再按ESC，然后输入冒号wq 即 :wq 就保存了。格式如下图：

```
root@localhost:~
ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAQEApa2HK0Zq0w1QDve4Gp18UqqT/TJMTjRIOd4Xcx/q+E7LjjM57phOE
VS9G811zT8A9Zffta+/dtGWZSgIif74vJEjvo9e93KncM0xNNvqGIKnTrqnrtBlCXlrLkjhvOF4kXnYRlEMJKz98n
v2Xb4GtyIi8IAtb9oZGq68M/3xmHlBBvDSPBN2BKoYXaDMaBt2BzqhuaFVQ83W7SUr60vj0XLrnAUHqq8LkGh3+Pv
qRb7IQ4w2E9a147WXlIxbRzLQFW7HroAJ3fB622A4H7VvPD95bZZ913xnb+HjBAuxti8uf3/ezm8Via7vzT90VPv2
pQqMvDlnkgDV65eADWu02w==
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

4）再设置putty选项，点窗口左侧的SSh –> Auth， 单击窗口右侧的Browse… 选择刚刚生成的私钥， 再点Open， 此时输入root，就不用输入密码就能登录了。



如果在前面你设置了Key Passphrase， 那么此时就会提示你输入密码的。为了更加安全建议大家要设置一个Key Passphrase。

Linux 文件基本属性

Linux系统是一种典型的多用户系统，不同的用户处于不同的地位，拥有不同的权限。为了保护系统的安全性，Linux系统对不同的用户访问同一文件（包括目录文件）的权限做了不同的规定。

在Linux中我们可以使用ll或者ls –l命令来显示一个文件的属性以及文件所属的用户和组，如：

```
[root@www /]# ls -l
total 64
dr-xr-xr-x   2 root root 4096 Dec 14  2012 bin
dr-xr-xr-x   4 root root 4096 Apr 19  2012 boot
……
```

实例中，bin文件的第一个属性用"d"表示。"d"在Linux中代表该文件是一个目录文件。

在Linux中第一个字符代表这个文件是目录、文件或链接文件等等。

- 当为[*d*]则是目录
- 当为[-]则是文件；
- 若是[*l*]则表示为链接文档(link file)；
- 若是[*b*]则表示为装置文件里面的可供储存的接口设备(可随机存取装置)；
- 若是[*c*]则表示为装置文件里面的串行端口设备，例如键盘、鼠标(一次性读取装置)。

接下来的字符中，以三个为一组，且均为『rwx』 的三个参数的组合。其中，[r]代表可读(read)、[w]代表可写(write)、[x]代表可执行(execute)。 要注意的是，这三个权限的位置不会改变，如果没有权限，就会出现减号[-]而已。

每个文件的属性由左边第一部分的10个字符来确定（如下图）。

文件类型	属主权限			属组权限			其他用户权限		
0	1	2	3	4	5	6	7	8	9
d	r	w	x	r	-	x	r	-	x
目录文件	读	写	执行	读	写	执行	读	写	执行

从左至右用0-9这些数字来表示。

第0位确定文件类型，第1-3位确定属主（该文件的所有者）拥有该文件的权限。

第4-6位确定属组（所有者的同组用户）拥有该文件的权限，第7-9位确定其他用户拥有该文件的权限。

其中，第1、4、7位表示读权限，如果用"r"字符表示，则有读权限，如果用"-"字符表示，则没有读权限；

第2、5、8位表示写权限，如果用"w"字符表示，则有写权限，如果用"-"字符表示没有写权限；第3、6、9位表示可执行权限，如果用"x"字符表示，则有执行权限，如果用"-"字符表示，则没有执行权限。

Linux文件属主和属组

```
[root@www /]# ls -l
total 64
dr-xr-xr-x   2 root root 4096 Dec 14  2012 bin
dr-xr-xr-x   4 root root 4096 Apr 19  2012 boot
……
```

对于文件来说，它都有一个特定的所有者，也就是对该文件具有所有权的用户。

同时，在Linux系统中，用户是按组分类的，一个用户属于一个或多个组。

文件所有者以外的用户又可以分为文件所有者的同组用户和其他用户。

因此，Linux系统按文件所有者、文件所有者同组用户和其他用户来规定了不同的文件访问权限。

在以上实例中，bin文件是一个目录文件，属主和属组都为root，属主有可读、可写、可执行的权限；与属主同组的其他用户有可读和可执行的权限；其他用户也有可读和可执行的权限。

更改文件属性

1、chgrp：更改文件属组

语法：

```
chgrp [-R] 属组名文件名
```

参数选项

- -R：递归更改文件属组，就是在更改某个目录文件的属组时，如果加上-R的参数，那么该目录下的所有文件的属组都会更改。

2、chown：更改文件属主，也可以同时更改文件属组

语法：

```
chown [-R] 属主名 文件名
chown [-R] 属主名：属组名 文件名
```

进入 /root 目录（~）将install.log的拥有者改为bin这个账号：

```
[root@www ~] cd ~
[root@www ~]# chown bin install.log
[root@www ~]# ls -l
-rw-r--r--  1 bin  users 68495 Jun 25 08:53 install.log
```

将install.log的拥有者与群组改回为root：

```
[root@www ~]# chown root:root install.log
[root@www ~]# ls -l
-rw-r--r--  1 root root 68495 Jun 25 08:53 install.log
```

3、 chmod：更改文件9个属性

Linux文件属性有两种设置方法，一种是数字，一种是符号。

Linux文件的基本权限就有九个，分别是owner/group/others三种身份各有自己的read/write/execute权限。

先复习一下刚刚上面提到的数据：文件的权限字符为：『-rwxrwxrwx』， 这九个权限是三个三个一组的！其中，我们可以使用数字来代表各个权限，各权限的分数对照表如下：

- r:4
- w:2
- x:1

每种身份(owner/group/others)各自的三个权限(r/w/x)分数是需要累加的，例如当权限为： [-rwxrwx---] 分数则是：

- owner = rwx = 4+2+1 = 7
- group = rwx = 4+2+1 = 7
- others= --- = 0+0+0 = 0

所以等一下我们设定权限的变更时，该文件的权限数字就是770啦！变更权限的指令chmod的语法是这样的：

```
 chmod [-R] xyz 文件或目录
```

选项与参数：

- xyz : 就是刚刚提到的数字类型的权限属性， 为 rwx 属性数值的相加。
- -R : 进行递归(recursive)的持续变更，亦即连同次目录下的所有文件都会变更

举例来说，如果要将.bashrc这个文件所有的权限都设定启用，那么命令如下：

```
[root@www ~]# ls -al .bashrc
-rw-r--r--  1 root root 395 Jul  4 11:45 .bashrc
[root@www ~]# chmod 777 .bashrc
[root@www ~]# ls -al .bashrc
-rwxrwxrwx  1 root root 395 Jul  4 11:45 .bashrc
```

那如果要将权限变成 *-rwxr-xr--* 呢？那么权限的分数就成为 [4+2+1][4+0+1][4+0+0]=754。

符号类型改变文件权限

还有一个改变权限的方法呦！从之前的介绍中我们可以发现，基本上就九个权限分别是(1)user (2)group (3)others三种身份啦！ 那么我们就可以藉由u, g, o来代表三种身份的权限！

此外， a 则代表 all 亦即全部的身份！那么读写的权限就可以写成r, w, x！也就是可以使用底下的方式来看：

chmod	u g o a	+(加入) -(除去) =(设定)	r w x	文件或目录

如果我们需要将文件权限设置为 -rwxr-xr-- ， 可以使用 *chmod u=rwx,g=rx,o=r* 文件名 来设定:

```
[root@www ~]# ls -al .bashrc
-rwxr-xr-x  1 root root 395 Jul  4 11:45 .bashrc
[root@www ~]# chmod  a+w  .bashrc
[root@www ~]# ls -al .bashrc
-rwxrwxrwx  1 root root 395 Jul  4 11:45 .bashrc
```

而如果是要将权限去掉而不改变其他已存在的权限呢？例如要拿掉全部人的可执行权限，则：

```
[root@www ~]# chmod  a-x  .bashrc
[root@www ~]# ls -al .bashrc
-rw-rw-rw-  1 root root 395 Jul  4 11:45 .bashrc
```

Linux 文件与目录管理

我们知道Linux的目录结构为树状结构，最顶级的目录为根目录 /。

其他目录通过挂载可以将它们添加到树中，通过解除挂载可以移除它们。

在开始本教程前我们需要先知道什么是绝对路径与相对路径。

- 绝对路径：
  路径的写法，由根目录 / 写起，例如： /usr/share/doc 这个目录。
- 相对路径：
  路径的写法，不是由 / 写起，例如由 /usr/share/doc 要到 /usr/share/man 底下时，可以写成： cd ../man 这就是相对路径的写法啦！

处理目录的常用命令

接下来我们就来看几个常见的处理目录的命令吧：

- ls: 列出目录
- cd：切换目录
- pwd：显示目前的目录
- mkdir：创建一个新的目录
- rmdir：删除一个空的目录
- cp: 复制文件或目录
- rm: 移除文件或目录

你可以使用 *man [命令]* 来查看各个命令的使用文档，如 ：man cp。

ls (列出目录)

在Linux系统当中， ls 命令可能是最常被运行的。

语法：

```
[root@www ~]# ls [-aAdfFhilnrRSt] 目录名称
[root@www ~]# ls [--color={never,auto,always}] 目录名称
[root@www ~]# ls [--full-time] 目录名称
```

选项与参数：

- -a ：全部的文件，连同隐藏档(开头为 . 的文件) 一起列出来(常用)
- -d ：仅列出目录本身，而不是列出目录内的文件数据(常用)

- -l ：长数据串列出，包含文件的属性与权限等等数据；(常用)

将家目录下的所有文件列出来(含属性与隐藏档)

```
[root@www ~]# ls -al ~
```

cd (切换目录)

cd是Change Directory的缩写，这是用来变换工作目录的命令。

语法：

```
 cd [相对路径或绝对路径]
```

```
#使用 mkdir 命令创建w3cschool.cc目录
[root@www ~]# mkdir w3cschool.cc

#使用绝对路径切换到w3cschool.cc目录
[root@www ~]# cd /root/w3cschool.cc/

#使用相对路径切换到w3cschool.cc目录
[root@www ~]# cd ./w3cschool.cc/

# 表示回到自己的家目录，亦即是 /root 这个目录
[root@www w3cschool.cc]# cd ~

# 表示去到目前的上一级目录，亦即是 /root 的上一级目录的意思；
[root@www ~]# cd ..
```

接下来大家多操作几次应该就可以很好的理解 cd 命令的。

pwd (显示目前所在的目录)

pwd是Print Working Directory的缩写，也就是显示目前所在目录的命令。

```
[root@www ~]# pwd [-P]
选项与参数：
-P  ：显示出确实的路径，而非使用连结 (link) 路径。

范例：单纯显示出目前的工作目录：
[root@www ~]# pwd
/root   <== 显示出目录啦~

范例：显示出实际的工作目录，而非连结档本身的目录名而已
[root@www ~]# cd /var/mail   <==注意，/var/mail是一个连结档
[root@www mail]# pwd
/var/mail         <==列出目前的工作目录
[root@www mail]# pwd -P
/var/spool/mail   <==怎么回事？有没有加 -P 差很多~
[root@www mail]# ls -ld /var/mail
lrwxrwxrwx 1 root root 10 Sep  4 17:54 /var/mail -> spool/mail
# 看到这里应该知道为啥了吧？因为 /var/mail 是连结档，连结到 /var/spool/mail
# 所以，加上 pwd -P 的选项后，会不以连结档的数据显示，而是显示正确的完整路径啊！
```

mkdir (创建新目录)

如果想要创建新的目录的话，那么就使用mkdir (make directory)吧。

语法：

```
mkdir [-mp] 目录名称
```

选项与参数：

- -m ：配置文件的权限喔！直接配置，不需要看默认权限 (umask) 的脸色～
- -p ：帮助你直接将所需要的目录(包含上一级目录)递回创建起来！

范例：请到/tmp底下尝试创建数个新目录看看：

```
[root@www ~]# cd /tmp
[root@www tmp]# mkdir test    <==创建一名为 test 的新目录
[root@www tmp]# mkdir test1/test2/test3/test4
mkdir: cannot create directory `test1/test2/test3/test4':
No such file or directory       <== 没办法直接创建此目录啊！
[root@www tmp]# mkdir -p test1/test2/test3/test4
```

加了这个 -p 的选项，可以自行帮你创建多层目录！

范例：创建权限为rwx--x--x的目录

```
[root@www tmp]# mkdir -m 711 test2
[root@www tmp]# ls -l
drwxr-xr-x  3 root  root 4096 Jul 18 12:50 test
drwxr-xr-x  3 root  root 4096 Jul 18 12:53 test1
drwx--x--x  2 root  root 4096 Jul 18 12:54 test2
```

上面的权限部分，如果没有加上 -m 来强制配置属性，系统会使用默认属性。

如果我们使用 -m ，如上例我们给予 -m 711 来给予新的目录 drwx--x--x 的权限。

rmdir (删除空的目录)

语法：

```
 rmdir [-p] 目录名称
```

选项与参数：

- **-p** ：连同上一级『空的』目录也一起删除

删除 w3cschool.cc 目录

```
[root@www tmp]# rmdir w3cschool.cc/
```

范例：将於mkdir范例中创建的目录(/tmp底下)删除掉！

```
[root@www tmp]# ls -l   <==看看有多少目录存在？
drwxr-xr-x  3 root  root 4096 Jul 18 12:50 test
drwxr-xr-x  3 root  root 4096 Jul 18 12:53 test1
drwx--x--x  2 root  root 4096 Jul 18 12:54 test2
[root@www tmp]# rmdir test   <==可直接删除掉，没问题
[root@www tmp]# rmdir test1  <==因为尚有内容，所以无法删除！
rmdir: `test1': Directory not empty
[root@www tmp]# rmdir -p test1/test2/test3/test4
[root@www tmp]# ls -l        <==您看看，底下的输出中test与test1不见了！
drwx--x--x  2 root  root 4096 Jul 18 12:54 test2
```

利用 -p 这个选项，立刻就可以将 test1/test2/test3/test4 一次删除。

不过要注意的是，这个 rmdir 仅能删除空的目录，你可以使用 rm 命令来删除非空目录。

cp (复制文件或目录)

cp 即拷贝文件和目录。

语法:

```
[root@www ~]# cp [-adfilprsu] 来源档(source) 目标档(destination)
[root@www ~]# cp [options] source1 source2 source3 .... directory
```

选项与参数：

- -a ：相当於 -pdr 的意思，至於 pdr 请参考下列说明；(常用) *-d ：若来源档为连结档的属性(link file)*，则复制连结档属性而非文件本身； -f ：为强制(force)的意思，若目标文件已经存在且无法开启，则移除后再尝试一次； *-i ：若目标档(destination)*已经存在时，在覆盖时会先询问动作的进行*(常用) -l* ：进行硬式连结(hard link)的连结档创建，而非复制文件本身； *-p ：连同文件的属性一起复制过去，而非使用默认属性(备份常用)*； -r ：递回持续复制，用於目录的复制行为；(常用) *-s ：复制成为符号连结档 (symbolic link)*，亦即『捷径』文件； -u ：若 destination 比 source 旧才升级 destination ！

用root身份，将家目录下的 .bashrc 复制到 /tmp 下，并更名为 bashr

```
[root@www ~]# cp ~/.bashrc /tmp/bashrc
[root@www ~]# cp -i ~/.bashrc /tmp/bashrc
cp: overwrite `/tmp/bashrc'? n  <==n不覆盖，y为覆盖
```

rm (移除文件或目录)

语法：

```
rm [-fir] 文件或目录
```

选项与参数：

- -f ：就是 force 的意思，忽略不存在的文件，不会出现警告信息；
- -i ：互动模式，在删除前会询问使用者是否动作
- -r ：递回删除啊！最常用在目录的删除了！这是非常危险的选项！！！

将刚刚在 cp 的范例中创建的 bashrc 删除掉！

```
[root@www tmp]# rm -i bashrc
rm: remove regular file `bashrc'? y
```

如果加上 -i 的选项就会主动询问喔，避免你删除到错误的档名！

mv (移动文件与目录，或修改名称)

语法：

```
[root@www ~]# mv [-fiu] source destination
[root@www ~]# mv [options] source1 source2 source3 .... directory
```

选项与参数：

- -f ：force 强制的意思，如果目标文件已经存在，不会询问而直接覆盖；
- -i ：若目标文件 (destination) 已经存在时，就会询问是否覆盖！
- -u ：若目标文件已经存在，且 source 比较新，才会升级 (update)

复制一文件，创建一目录，将文件移动到目录中

```
[root@www ~]# cd /tmp
[root@www tmp]# cp ~/.bashrc bashrc
[root@www tmp]# mkdir mvtest
[root@www tmp]# mv bashrc mvtest
```

将某个文件移动到某个目录去，就是这样做！

将刚刚的目录名称更名为 mvtest2

```
[root@www tmp]# mv mvtest mvtest2
```

Linux 文件内容查看

Linux系统中使用以下命令来查看文件的内容：

- cat 由第一行开始显示文件内容
- tac 从最后一行开始显示，可以看出 tac 是 cat 的倒著写！
- nl 显示的时候，顺道输出行号！
- more 一页一页的显示文件内容
- less 与 more 类似，但是比 more 更好的是，他可以往前翻页！
- head 只看头几行
- tail 只看尾巴几行

你可以使用 *man [命令]*来查看各个命令的使用文档，如 ：man cp。

cat

由第一行开始显示文件内容

语法：

```
cat [-AbEnTv]
```

选项与参数：

- -A ：相当於 -vET 的整合选项，可列出一些特殊字符而不是空白而已；
- -b ：列出行号，仅针对非空白行做行号显示，空白行不标行号！
- -E ：将结尾的断行字节 $ 显示出来；
- -n ：列印出行号，连同空白行也会有行号，与 -b 的选项不同；
- -T ：将 [tab] 按键以 ^I 显示出来；
- -v ：列出一些看不出来的特殊字符

检看 /etc/issue 这个文件的内容：

```
[root@www ~]# cat /etc/issue
CentOS release 6.4 (Final)
Kernel \r on an \m
```

tac

tac与cat命令刚好相反，文件内容从最后一行开始显示，可以看出 tac 是 cat 的倒着写！如：

```
[root@www ~]# tac /etc/issue

Kernel \r on an \m
CentOS release 6.4 (Final)
```

nl

显示行号

语法：

```
nl [-bnw] 文件
```

选项与参数：

- -b ：指定行号指定的方式，主要有两种：
  -b a ：表示不论是否为空行，也同样列出行号(类似 cat -n)；
  -b t ：如果有空行，空的那一行不要列出行号(默认值)；
- -n ：列出行号表示的方法，主要有三种：
  -n ln ：行号在萤幕的最左方显示；
  -n rn ：行号在自己栏位的最右方显示，且不加 0 ；
  -n rz ：行号在自己栏位的最右方显示，且加 0 ；
- -w ：行号栏位的占用的位数。

范例一：用 nl 列出 /etc/issue 的内容

```
[root@www ~]# nl /etc/issue
     1  CentOS release 6.4 (Final)
     2  Kernel \r on an \m
```

more

一页一页翻动

```
[root@www ~]# more /etc/man.config
#
# Generated automatically from man.conf.in by the
# configure script.
#
# man.conf from man-1.6d
....(中间省略)....
--More--(28%)  <== 重点在这一行喔！你的光标也会在这里等待你的命令
```

在 more 这个程序的运行过程中，你有几个按键可以按的：

- 空白键 (space)：代表向下翻一页；
- Enter ：代表向下翻『一行』；
- /字串 ：代表在这个显示的内容当中，向下搜寻『字串』这个关键字；
- :f ：立刻显示出档名以及目前显示的行数；
- q ：代表立刻离开 more ，不再显示该文件内容。
- b 或 [ctrl]-b ：代表往回翻页，不过这动作只对文件有用，对管线无用。

less

一页一页翻动，以下实例输出/etc/man.config文件的内容：

```
[root@www ~]# less /etc/man.config
#
# Generated automatically from man.conf.in by the
# configure script.
#
# man.conf from man-1.6d
....(中间省略)....
:   <== 这里可以等待你输入命令！
```

less运行时可以输入的命令有：

- 空白键 ：向下翻动一页；
- [pagedown]：向下翻动一页；
- [pageup] ：向上翻动一页；
- /字串 ：向下搜寻『字串』的功能；
- ?字串 ：向上搜寻『字串』的功能；
- n ：重复前一个搜寻 (与 / 或 ? 有关！)
- N ：反向的重复前一个搜寻 (与 / 或 ? 有关！)
- q ：离开 less 这个程序；

head

取出文件前面几行

语法：

```
head [-n number] 文件
```

选项与参数：

- -n ：后面接数字，代表显示几行的意思

```
[root@www ~]# head /etc/man.config
```

默认的情况中，显示前面 10 行！若要显示前 20 行，就得要这样：

```
[root@www ~]# head -n 20 /etc/man.config
```

tail

取出文件后面几行

语法：

```
tail [-n number] 文件
```

选项与参数：

- -n ：后面接数字，代表显示几行的意思
- -f ：表示持续侦测后面所接的档名，要等到按下[ctrl]-c才会结束tail的侦测

```
[root@www ~]# tail /etc/man.config
# 默认的情况中，显示最后的十行！若要显示最后的 20 行，就得要这样：
[root@www ~]# tail -n 20 /etc/man.config
```

Linux 用户和用户组管理

Linux系统是一个多用户多任务的分时操作系统，任何一个要使用系统资源的用户，都必须首先向系统管理员申请一个账号，然后以这个账号的身份进入系统。

用户的账号一方面可以帮助系统管理员对使用系统的用户进行跟踪，并控制他们对系统资源的访问；另一方面也可以帮助用户组织文件，并为用户提供安全性保护。

每个用户账号都拥有一个惟一的用户名和各自的口令。

用户在登录时键入正确的用户名和口令后，就能够进入系统和自己的主目录。

实现用户账号的管理，要完成的工作主要有如下几个方面：

- 用户账号的添加、删除与修改。
- 用户口令的管理。
- 用户组的管理。

一、Linux系统用户账号的管理

用户账号的管理工作主要涉及到用户账号的添加、修改和删除。

添加用户账号就是在系统中创建一个新账号，然后为新账号分配用户号、用户组、主目录和登录Shell等资源。刚添加的账号是被锁定的，无法使用。

1、添加新的用户账号使用useradd命令，其语法如下：

```
useradd 选项 用户名
```

参数说明：

- 选项:
    - -c comment 指定一段注释性描述。
    - -d 目录 指定用户主目录，如果此目录不存在，则同时使用-m选项，可以创建主目录。
    - -g 用户组 指定用户所属的用户组。
    - -G 用户组，用户组 指定用户所属的附加组。
    - -s Shell文件 指定用户的登录Shell。
    - -u 用户号 指定用户的用户号，如果同时有-o选项，则可以重复使用其他用户的标识号。
- 用户名:

指定新账号的登录名。

实例1

```
# useradd –d /usr/sam -m sam
```

此命令创建了一个用户sam，其中-d和-m选项用来为登录名sam产生一个主目录/usr/sam（/usr为默认的用户主目录所在的父目录）。

实例2

```
# useradd -s /bin/sh -g group –G adm,root gem
```

此命令新建了一个用户gem，该用户的登录Shell是 `/bin/sh`， 它属于group用户组，同时又属于adm和root用户组，其中group用户组是其主组。

这里可能新建组：`#groupadd group及groupadd adm`

增加用户账号就是在/etc/passwd文件中为新用户增加一条记录，同时更新其他系统文件如/etc/shadow, /etc/group等。

Linux提供了集成的系统管理工具userconf，它可以用来对用户账号进行统一管理。

3、删除帐号

如果一个用户的账号不再使用，可以从系统中删除。删除用户账号就是要将/etc/passwd等系统文件中的该用户记录删除，必要时还删除用户的主目录。

删除一个已有的用户账号使用`userdel`命令，其格式如下：

```
userdel 选项 用户名
```

常用的选项是-r，它的作用是把用户的主目录一起删除。

例如：

```
# userdel sam
```

此命令删除用户sam在系统文件中（主要是/etc/passwd, /etc/shadow, /etc/group等）的记录，同时删除用户的主目录。

4、修改帐号

修改用户账号就是根据实际情况更改用户的有关属性，如用户号、主目录、用户组、登录Shell等。

修改已有用户的信息使用 `usermod` 命令，其格式如下：

```
usermod 选项 用户名
```

常用的选项包括 `-c, -d, -m, -g, -G, -s, -u以及-o等` ，这些选项的意义与 `useradd` 命令中的选项一样，可以为用户指定新的资源值。

另外，有些系统可以使用选项：-l 新用户名

这个选项指定一个新的账号，即将原来的用户名改为新的用户名。

例如：

```
# usermod -s /bin/ksh -d /home/z –g developer sam
```

此命令将用户sam的登录Shell修改为ksh，主目录改为/home/z，用户组改为developer。

5、用户口令的管理

用户管理的一项重要内容是用户口令的管理。用户账号刚创建时没有口令，但是被系统锁定，无法使用，必须为其指定口令后才可以使用，即使是指定空口令。

指定和修改用户口令的Shell命令是 `passwd` 。超级用户可以为自己和其他用户指定口令，普通用户只能用它修改自己的口令。命令的格式为：

```
passwd 选项 用户名
```

可使用的选项：

- -l 锁定口令，即禁用账号。
- -u 口令解锁。
- -d 使账号无口令。
- -f 强迫用户下次登录时修改口令。

如果默认用户名，则修改当前用户的口令。

例如，假设当前用户是sam，则下面的命令修改该用户自己的口令：

```
$ passwd
Old password:******
New password:*******
Re-enter new password:*******
```

如果是超级用户，可以用下列形式指定任何用户的口令：

```
# passwd sam
New password:*******
Re-enter new password:*******
```

普通用户修改自己的口令时，passwd命令会先询问原口令，验证后再要求用户输入两遍新口令，如果两次输入的口令一致，则将这个口令指定给用户；而超级用户为用户指定口令时，就不需要知道原口令。

为了系统安全起见，用户应该选择比较复杂的口令，例如最好使用8位长的口令，口令中包含有大写、小写字母和数字，并且应该与姓名、生日等不相同。

为用户指定空口令时，执行下列形式的命令：

```
# passwd -d sam
```

此命令将用户sam的口令删除，这样用户sam下一次登录时，系统就不再询问口令。

passwd命令还可以用-l(lock)选项锁定某一用户，使其不能登录，例如：

```
# passwd -l sam
```

二、Linux系统用户组的管理

每个用户都有一个用户组，系统可以对一个用户组中的所有用户进行集中管理。不同Linux 系统对用户组的规定有所不同，如Linux下的用户属于与它同名的用户组，这个用户组在创建用户时同时创建。

用户组的管理涉及用户组的添加、删除和修改。组的增加、删除和修改实际上就是对/etc/group文件的更新。

1、增加一个新的用户组使用groupadd命令。其格式如下：

```
groupadd 选项 用户组
```

可以使用的选项有：

- -g GID 指定新用户组的组标识号（GID）。
- -o 一般与-g选项同时使用，表示新用户组的GID可以与系统已有用户组的GID相同。

实例1：

```
# groupadd group1
```

此命令向系统中增加了一个新组group1，新组的组标识号是在当前已有的最大组标识号的基础上加1。

实例**2**：

```
# groupadd -g 101 group2
```

此命令向系统中增加了一个新组group2，同时指定新组的组标识号是101。

2、如果要删除一个已有的用户组，使用groupdel命令，其格式如下：

```
groupdel 用户组
```

例如：

```
# groupdel group1
```

此命令从系统中删除组group1。

3、修改用户组的属性使用groupmod命令。其语法如下：

```
groupmod 选项 用户组
```

常用的选项有：

- -g GID 为用户组指定新的组标识号。
- -o 与-g选项同时使用，用户组的新GID可以与系统已有用户组的GID相同。
- -n新用户组 将用户组的名字改为新名字

实例**1**：

```
# groupmod -g 102 group2
```

此命令将组group2的组标识号修改为102。

实例**2**：

```
# groupmod –g 10000 -n group3 group2
```

此命令将组group2的标识号改为10000，组名修改为group3。

4、如果一个用户同时属于多个用户组，那么用户可以在用户组之间切换，以便具有其他用户组的权限。

用户可以在登录后，使用命令newgrp切换到其他用户组，这个命令的参数就是目的用户组。例如：

```
$ newgrp root
```

这条命令将当前用户切换到root用户组，前提条件是root用户组确实是该用户的主组或附加组。类似于用户账号的管理，用户组的管理也可以通过集成的系统管理工具来完成。

三、与用户账号有关的系统文件

完成用户管理的工作有许多种方法，但是每一种方法实际上都是对有关的系统文件进行修改。

与用户和用户组相关的信息都存放在一些系统文件中，这些文件包括/etc/passwd, /etc/shadow, /etc/group等。

下面分别介绍这些文件的内容。

1、/etc/passwd文件是用户管理工作涉及的最重要的一个文件。

Linux系统中的每个用户都在/etc/passwd文件中有一个对应的记录行，它记录了这个用户的一些基本属性。

这个文件对所有用户都是可读的。它的内容类似下面的例子：

```
# cat /etc/passwd

root:x:0:0:Superuser:/:
daemon:x:1:1:System daemons:/etc:
bin:x:2:2:Owner of system commands:/bin:
sys:x:3:3:Owner of system files:/usr/sys:
adm:x:4:4:System accounting:/usr/adm:
uucp:x:5:5:UUCP administrator:/usr/lib/uucp:
auth:x:7:21:Authentication administrator:/tcb/files/auth:
cron:x:9:16:Cron daemon:/usr/spool/cron:
listen:x:37:4:Network daemon:/usr/net/nls:
lp:x:71:18:Printer administrator:/usr/spool/lp:
sam:x:200:50:Sam san:/usr/sam:/bin/sh
```

从上面的例子我们可以看到，/etc/passwd中一行记录对应着一个用户，每行记录又被冒号(:)分隔为7个字段，其格式和具体含义如下：

```
用户名:口令:用户标识号:组标识号:注释性描述:主目录:登录Shell
```

1）"用户名"是代表用户账号的字符串。

通常长度不超过8个字符，并且由大小写字母和/或数字组成。登录名中不能有冒号(:)，因为冒号在这里是分隔符。

为了兼容起见，登录名中最好不要包含点字符(.)，并且不使用连字符(-)和加号(+)打头。

2）“口令”一些系统中，存放着加密后的用户口令字。

虽然这个字段存放的只是用户口令的加密串，不是明文，但是由于/etc/passwd文件对所有用户都可读，所以这仍是一个安全隐患。因此，现在许多Linux 系统（如SVR4）都使用了shadow技术，把真正的加密后的用户口令字存放到/etc/shadow文件中，而在/etc/passwd文件的口令字段中只存放一个特殊的字符，例如“x”或者“*”。

3）“用户标识号”是一个整数，系统内部用它来标识用户。

一般情况下它与用户名是一一对应的。如果几个用户名对应的用户标识号是一样的，系统内部将把它们视为同一个用户，但是它们可以有不同的口令、不同的主目录以及不同的登录Shell等。

通常用户标识号的取值范围是0～65 535。0是超级用户root的标识号，1～99由系统保留，作为管理账号，普通用户的标识号从100开始。在Linux系统中，这个界限是500。

4）“组标识号”字段记录的是用户所属的用户组。

它对应着/etc/group文件中的一条记录。

5)“注释性描述”字段记录着用户的一些个人情况。

例如用户的真实姓名、电话、地址等，这个字段并没有什么实际的用途。在不同的Linux 系统中，这个字段的格式并没有统一。在许多Linux系统中，这个字段存放的是一段任意的注释性描述文字，用做finger命令的输出。

6)“主目录”，也就是用户的起始工作目录。

它是用户在登录到系统之后所处的目录。在大多数系统中，各用户的主目录都被组织在同一个特定的目录下，而用户主目录的名称就是该用户的登录名。各用户对自己的主目录有读、写、执行（搜索）权限，其他用户对此目录的访问权限则根据具体情况设置。

7)用户登录后，要启动一个进程，负责将用户的操作传给内核，这个进程是用户登录到系统后运行的命令解释器或某个特定的程序，即Shell。

Shell是用户与Linux系统之间的接口。Linux的Shell有许多种，每种都有不同的特点。常用的有sh(Bourne Shell), csh(C Shell), ksh(Korn Shell), tcsh(TENEX/TOPS-20 type C Shell), bash(Bourne Again Shell)等。

系统管理员可以根据系统情况和用户习惯为用户指定某个Shell。如果不指定Shell，那么系统使用sh为默认的登录Shell，即这个字段的值为/bin/sh。

用户的登录Shell也可以指定为某个特定的程序（此程序不是一个命令解释器）。

利用这一特点，我们可以限制用户只能运行指定的应用程序，在该应用程序运行结束后，用户就自动退出了系统。有些Linux 系统要求只有那些在系统中登记了的程序才能出现在这个字段中。

8)系统中有一类用户称为伪用户（psuedo users）。

这些用户在/etc/passwd文件中也占有一条记录，但是不能登录，因为它们的登录Shell为空。它们的存在主要是方便系统管理，满足相应的系统进程对文件属主的要求。

常见的伪用户如下所示：

```
伪 用 户 含 义
bin 拥有可执行的用户命令文件
sys 拥有系统文件
adm 拥有帐户文件
uucp UUCP使用
lp lp或lpd子系统使用
nobody NFS使用
```

拥有帐户文件

1、除了上面列出的伪用户外，还有许多标准的伪用户，例如：audit, cron, mail, usenet等，它们也都各自为相关的进程和文件所需要。

由于/etc/passwd文件是所有用户都可读的，如果用户的密码太简单或规律比较明显的话，一台普通的计算机就能够很容易地将它破解，因此对安全性要求较高的Linux系统都把加密后的口令字分离出来，单独存放在一个文件中，这个文件是/etc/shadow文件。 有超级用户才拥有该文件读权限，这就保证了用户密码的安全性。

2、/etc/shadow中的记录行与/etc/passwd中的一一对应，它由pwconv命令根据/etc/passwd中的数据自动产生

它的文件格式与/etc/passwd类似，由若干个字段组成，字段之间用":"隔开。这些字段是：

```
登录名:加密口令:最后一次修改时间:最小时间间隔:最大时间间隔:警告时间:不活动时间:失效时间:标志
```

1. "登录名"是与/etc/passwd文件中的登录名相一致的用户账号
2. "口令"字段存放的是加密后的用户口令字，长度为13个字符。如果为空，则对应用户没有口令，登录时不需要口令；如果含有不属于集合 { ./0-9A-Za-z }中的字符，则对应的用户不能登录。
3. "最后一次修改时间"表示的是从某个时刻起，到用户最后一次修改口令时的天数。时间起点对不同的系统可能不一样。例如在SCO Linux 中，这个时间起点是1970年1月1日。
4. "最小时间间隔"指的是两次修改口令之间所需的最小天数。
5. "最大时间间隔"指的是口令保持有效的最大天数。
6. "警告时间"字段表示的是从系统开始警告用户到用户密码正式失效之间的天数。
7. "不活动时间"表示的是用户没有登录活动但账号仍能保持有效的最大天数。
8. "失效时间"字段给出的是一个绝对的天数，如果使用了这个字段，那么就给出相应账号的生存期。期满后，该账号就不再是一个合法的账号，也就不能再用来登录了。

下面是/etc/shadow的一个例子：

```
 # cat /etc/shadow

root:Dnakfw28zf38w:8764:0:168:7:::
daemon:*::0:0::::
bin:*::0:0::::
sys:*::0:0::::
adm:*::0:0::::
uucp:*::0:0::::
nuucp:*::0:0::::
auth:*::0:0::::
cron:*::0:0::::
listen:*::0:0::::
lp:*::0:0::::
sam:EkdiSECLWPdSa:9740:0:0::::
```

3、用户组的所有信息都存放在/etc/group文件中。

将用户分组是Linux 系统中对用户进行管理及控制访问权限的一种手段。

每个用户都属于某个用户组；一个组中可以有多个用户，一个用户也可以属于不同的组。

当一个用户同时是多个组中的成员时，在/etc/passwd文件中记录的是用户所属的主组，也就是登录时所属的默认组，而其他组称为附加组。

用户要访问属于附加组的文件时，必须首先使用newgrp命令使自己成为所要访问的组中的成员。

用户组的所有信息都存放在/etc/group文件中。此文件的格式也类似于/etc/passwd文件，由冒号(:)隔开若干个字段，这些字段有：

```
组名:口令:组标识号:组内用户列表
```

1. "组名"是用户组的名称，由字母或数字构成。与/etc/passwd中的登录名一样，组名不应重复。
2. "口令"字段存放的是用户组加密后的口令字。一般Linux 系统的用户组都没有口令，即这个字段一般为空，或者是*。
3. "组标识号"与用户标识号类似，也是一个整数，被系统内部用来标识组。
4. "组内用户列表"是属于这个组的所有用户的列表/b]，不同用户之间用逗号(,)分隔。这个用户组可能是用户的主组，也可能是附加组。

/etc/group文件的一个例子如下：

```
root::0:root
bin::2:root,bin
sys::3:root,uucp
adm::4:root,adm
daemon::5:root,daemon
lp::7:root,lp
users::20:root,sam
```

四、添加量用户批

添加和删除用户对每位Linux系统管理员都是轻而易举的事，比较棘手的是如果要添加几十个、上百个甚至上千个用户时，我们不太可能还使用useradd一个一个地添加，必然要找一种简便的创建大量用户的方法。Linux系统提供了创建大量用户的工具，可以让您立即创建大量用户，方法如下：

（1）先编辑一个文本用户文件。

每一列按照 `/etc/passwd` 密码文件的格式书写，要注意每个用户的用户名、UID、宿主目录都不可以相同，其中密码栏可以留做空白或输入x号。一个范例文件user.txt内容如下：

```
user001::600:100:user:/home/user001:/bin/bash
user002::601:100:user:/home/user002:/bin/bash
user003::602:100:user:/home/user003:/bin/bash
user004::603:100:user:/home/user004:/bin/bash
user005::604:100:user:/home/user005:/bin/bash
user006::605:100:user:/home/user006:/bin/bash
```

（2）以root身份执行命令 `/usr/sbin/newusers`，从刚创建的用户文件 `user.txt` 中导入数据，创建用户：

```
# newusers < user.txt
```

然后可以执行命令 `vipw` 或 `vi /etc/passwd` 检查 `/etc/passwd` 文件是否已经出现这些用户的数据，并且用户的宿主目录是否已经创建。

（3）执行命令/usr/sbin/pwunconv。

将 `/etc/shadow` 产生的 `shadow` 密码解码，然后回写到 `/etc/passwd` 中，并将 `/etc/shadow` 的 `shadow` 密码栏删掉。这是为了方便下一步的密码转换工作，即先取消 `shadow password` 功能。

```
# pwunconv
```

（4）编辑每个用户的密码对照文件。

范例文件 `passwd.txt` 内容如下：

```
user001:密码
user002:密码
user003:密码
user004:密码
user005:密码
user006:密码
```

（5）以root身份执行命令 `/usr/sbin/chpasswd`。

创建用户密码，`chpasswd` 会将经过 `/usr/bin/passwd` 命令编码过的密码写入 `/etc/passwd` 的密码栏。

```
# chpasswd < passwd.txt
```

（6）确定密码经编码写入/etc/passwd的密码栏后。

执行命令 `/usr/sbin/pwconv` 将密码编码为 `shadow password`，并将结果写入 `/etc/shadow`。

```
# pwconv
```

这样就完成了大量用户的创建了，之后您可以到/home下检查这些用户宿主目录的权限设置是否都正确，并登录验证用户密码是否正确。

Linux 磁盘管理

Linux磁盘管理好坏管理直接关系到整个系统的性能问题。

Linux磁盘管理常用三个命令为df、du和fdisk。

- df：列出文件系统的整体磁盘使用量
- du：检查磁盘空间使用量
- fdisk：用于磁盘分区

df

df命令参数功能：检查文件系统的磁盘空间占用情况。可以利用该命令来获取硬盘被占用了多少空间，目前还剩下多少空间等信息。

语法：

```
df [-ahikHTm] [目录或文件名]
```

选项与参数：

- -a ：列出所有的文件系统，包括系统特有的 /proc 等文件系统；
- -k ：以 KBytes 的容量显示各文件系统；
- -m ：以 MBytes 的容量显示各文件系统；
- -h ：以人们较易阅读的 GBytes, MBytes, KBytes 等格式自行显示；
- -H ：以 M=1000K 取代 M=1024K 的进位方式；
- -T ：显示文件系统类型, 连同该 partition 的 filesystem 名称 (例如 ext3) 也列出；
- -i ：不用硬盘容量，而以 inode 的数量来显示

实例 1

将系统内所有的文件系统列出来！

```
[root@www ~]# df
Filesystem      1K-blocks      Used Available Use% Mounted on
/dev/hdc2         9920624   3823112   5585444  41% /
/dev/hdc3         4956316    141376   4559108   4% /home
/dev/hdc1          101086     11126     84741  12% /boot
tmpfs              371332         0    371332   0% /dev/shm
```

在 Linux 底下如果 df 没有加任何选项，那么默认会将系统内所有的 (不含特殊内存内的文件系统与 swap) 都以 1 Kbytes 的容量来列出来！

实例 2

将容量结果以易读的容量格式显示出来

```
[root@www ~]# df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/hdc2             9.5G  3.7G  5.4G  41% /
/dev/hdc3             4.8G  139M  4.4G   4% /home
/dev/hdc1              99M   11M   83M  12% /boot
tmpfs                 363M     0  363M   0% /dev/shm
```

实例 3

将系统内的所有特殊文件格式及名称都列出来

```
[root@www ~]# df -aT
Filesystem    Type 1K-blocks    Used Available Use% Mounted on
/dev/hdc2     ext3   9920624 3823112   5585444  41% /
proc          proc         0       0         0   -  /proc
sysfs        sysfs         0       0         0   -  /sys
devpts      devpts         0       0         0   -  /dev/pts
/dev/hdc3     ext3   4956316  141376   4559108   4% /home
/dev/hdc1     ext3    101086   11126     84741  12% /boot
tmpfs        tmpfs    371332       0    371332   0% /dev/shm
none   binfmt_misc         0       0         0   -  /proc/sys/fs/binfmt_misc
sunrpc  rpc_pipefs         0       0         0   -  /var/lib/nfs/rpc_pipefs
```

实例 4

将 /etc 底下的可用的磁盘容量以易读的容量格式显示

```
[root@www ~]# df -h /etc
Filesystem            Size  Used Avail Use% Mounted on
/dev/hdc2             9.5G  3.7G  5.4G  41% /
```

du

inux du命令也是查看使用空间的，但是与df命令不同的是Linux du命令是对文件和目录磁盘使用的空间的查看，还是和df命令有一些区别的，这里介绍Linux du命令。

语法：

```
du [-ahskm] 文件或目录名称
```

选项与参数：

- -a ：列出所有的文件与目录容量，因为默认仅统计目录底下的文件量而已。
- -h ：以人们较易读的容量格式 (G/M) 显示；

- -s ：列出总量而已，而不列出每个各别的目录占用容量；
- -S ：不包括子目录下的总计，与 -s 有点差别。
- -k ：以 KBytes 列出容量显示；
- -m ：以 MBytes 列出容量显示；

实例 1

列出目前目录下的所有文件容量

```
[root@www ~]# du
8       ./test4     <==每个目录都会列出来
8       ./test2
....中间省略....
12      ./.gconfd   <==包括隐藏文件的目录
220     .           <==这个目录(.)所占用的总量
```

直接输入 du 没有加任何选项时，则 du 会分析当前所在目录的文件与目录所占用的硬盘空间。

实例 2

将文件的容量也列出来

```
[root@www ~]# du -a
12      ./install.log.syslog   <==有文件的列表了
8       ./.bash_logout
8       ./test4
8       ./test2
....中间省略....
12      ./.gconfd
220     .
```

实例 3

检查根目录底下每个目录所占用的容量

```
[root@www ~]# du -sm /*
7       /bin
6       /boot
.....中间省略....
0       /proc
.....中间省略....
1       /tmp
3859    /usr     <==系统初期最大就是他了啦！
77      /var
```

通配符 * 来代表每个目录。

与 df 不一样的是，du 这个命令其实会直接到文件系统内去搜寻所有的文件数据。

fdisk

fdisk 是 Linux 的磁盘分区表操作工具。

语法：

```
fdisk [-l] 装置名称
```

选项与参数：

- -l ：输出后面接的装置所有的分区内容。若仅有 fdisk -l 时， 则系统将会把整个系统内能够搜寻到的装置的分区均列出来。

实例 1

列出所有分区信息

```
[root@AY120919111755c246621 tmp]# fdisk -l

Disk /dev/xvda: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

    Device Boot      Start         End      Blocks   Id  System
/dev/xvda1   *           1        2550    20480000   83  Linux
/dev/xvda2            2550        2611      490496   82  Linux swap / Solaris

Disk /dev/xvdb: 21.5 GB, 21474836480 bytes
255 heads, 63 sectors/track, 2610 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x56f40944

    Device Boot      Start         End      Blocks   Id  System
/dev/xvdb2               1        2610    20964793+  83  Linux
```

实例 2

找出你系统中的根目录所在磁盘，并查阅该硬盘内的相关信息

```
[root@www ~]# df /            <==注意：重点在找出磁盘文件名而已
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hdc2              9920624   3823168   5585388  41% /

[root@www ~]# fdisk /dev/hdc  <==仔细看，不要加上数字喔！
The number of cylinders for this disk is set to 5005.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):     <==等待你的输入！
```

输入 m 后，就会看到底下这些命令介绍

```
Command (m for help): m   <== 输入 m 后，就会看到底下这些命令介绍
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition            <==删除一个partition
   l   list known partition types
   m   print this menu
   n   add a new partition           <==新增一个partition
   o   create a new empty DOS partition table
   p   print the partition table     <==在屏幕上显示分割表
   q   quit without saving changes   <==不储存离开fdisk程序
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit  <==将刚刚的动作写入分割表
   x   extra functionality (experts only)
```

离开 fdisk 时按下 `q`，那么所有的动作都不会生效！相反的， 按下 `w` 就是动作生效的意思。

```
Command (m for help): p  <== 这里可以输出目前磁盘的状态

Disk /dev/hdc: 41.1 GB, 41174138880 bytes        <==这个磁盘的文件名与容量
255 heads, 63 sectors/track, 5005 cylinders      <==磁头、扇区与磁柱大小
Units = cylinders of 16065 * 512 = 8225280 bytes <==每个磁柱的大小

   Device Boot      Start         End      Blocks   Id  System
/dev/hdc1   *           1          13      104391   83  Linux
/dev/hdc2              14        1288    10241437+  83  Linux
/dev/hdc3            1289        1925     5116702+  83  Linux
/dev/hdc4            1926        5005    24740100    5  Extended
/dev/hdc5            1926        2052     1020096   82  Linux swap / Solaris
# 装置文件名 启动区否 开始磁柱    结束磁柱  1K大小容量 磁盘分区槽内的系统

Command (m for help): q
```

想要不储存离开吗？按下 q 就对了！不要随便按 w 啊！

使用 `p` 可以列出目前这颗磁盘的分割表信息，这个信息的上半部在显示整体磁盘的状态。

磁盘格式化

磁盘分割完毕后自然就是要进行文件系统的格式化，格式化的命令非常的简单，使用 `mkfs`（make filesystem） 命令。

语法：

```
mkfs [-t 文件系统格式] 装置文件名
```

选项与参数：

- -t ：可以接文件系统格式，例如 ext3, ext2, vfat 等(系统有支持才会生效)

实例 1

查看 mkfs 支持的文件格式

```
[root@www ~]# mkfs[tab][tab]
mkfs         mkfs.cramfs  mkfs.ext2    mkfs.ext3    mkfs.msdos   mkfs.vfat
```

按下两个[tab]，会发现 mkfs 支持的文件格式如上所示。

实例 2

将分区 /dev/hdc6（可指定你自己的分区） 格式化为 ext3 文件系统：

```
[root@www ~]# mkfs -t ext3 /dev/hdc6
mke2fs 1.39 (29-May-2006)
Filesystem label=                <==这里指的是分割槽的名称(label)
OS type: Linux
Block size=4096 (log=2)          <==block 的大小配置为 4K
Fragment size=4096 (log=2)
251392 inodes, 502023 blocks     <==由此配置决定的inode/block数量
25101 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=515899392
16 block groups
32768 blocks per group, 32768 fragments per group
15712 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Writing inode tables: done
Creating journal (8192 blocks): done <==有日志记录
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 34 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
# 这样就创建起来我们所需要的 Ext3 文件系统了！简单明了！
```

磁盘检验

fsck（file system check）用来检查和维护不一致的文件系统。

若系统掉电或磁盘发生问题，可利用fsck命令对文件系统进行检查。

语法：

```
fsck [-t 文件系统] [-ACay] 装置名称
```

选项与参数：

- -t : 给定档案系统的型式，若在 /etc/fstab 中已有定义或 kernel 本身已支援的则不需加上此参数
- -s : 依序一个一个地执行 fsck 的指令来检查
- -A : 对/etc/fstab 中所有列出来的 分区（partition）做检查
- -C : 显示完整的检查进度
- -d : 打印出 e2fsck 的 debug 结果
- -p : 同时有 -A 条件时，同时有多个 fsck 的检查一起执行
- -R : 同时有 -A 条件时，省略 / 不检查
- -V : 详细显示模式
- -a : 如果检查有错则自动修复
- -r : 如果检查有错则由使用者回答是否修复
- -y : 选项指定检测每个文件是自动输入yes，在不确定那些是不正常的时候，可以执行 # fsck -y 全部检查修复。

实例 1

查看系统有多少文件系统支持的 fsck 命令：

```
[root@www ~]# fsck[tab][tab]
fsck          fsck.cramfs  fsck.ext2    fsck.ext3    fsck.msdos   fsck.vfat
```

实例 2

强制检测 /dev/hdc6 分区:

```
[root@www ~]# fsck -C -f -t ext3 /dev/hdc6
fsck 1.39 (29-May-2006)
e2fsck 1.39 (29-May-2006)
Pass 1: Checking inodes, blocks, and sizes
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
vbird_logical: 11/251968 files (9.1% non-contiguous), 36926/1004046 blocks
```

如果没有加上 -f 的选项，则由于这个文件系统不曾出现问题，检查的经过非常快速！若加上 -f 强制检查，才会一项一项的显示过程。

磁盘挂载与卸除

Linux 的磁盘挂载使用 `mount` 命令，卸载使用 `umount` 命令。

磁盘挂载语法：

```
mount [-t 文件系统] [-L Label名] [-o 额外选项] [-n]  装置文件名  挂载点
```

实例 1

用默认的方式，将刚刚创建的 /dev/hdc6 挂载到 /mnt/hdc6 上面！

```
[root@www ~]# mkdir /mnt/hdc6
[root@www ~]# mount /dev/hdc6 /mnt/hdc6
[root@www ~]# df
Filesystem           1K-blocks      Used Available Use% Mounted on
.....中间省略.....
/dev/hdc6              1976312     42072   1833836   3% /mnt/hdc6
```

磁盘卸载命令 `umount` 语法：

```
umount [-fn] 装置文件名或挂载点
```

选项与参数：

- -f ：强制卸除！可用在类似网络文件系统 (NFS) 无法读取到的情况下；
- -n ：不升级 /etc/mtab 情况下卸除。

卸载/dev/hdc6

```
[root@www ~]# umount /dev/hdc6
```

Linux vi/vim

所有的 Unix Like 系统都会内建 vi 文书编辑器，其他的文书编辑器则不一定会存在。

但是目前我们使用比较多的是 vim 编辑器。

vim 具有程序编辑的能力，可以主动的以字体颜色辨别语法的正确性，方便程序设计。

什么是 vim？

Vim是从 vi 发展出来的一个文本编辑器。代码补完、编译及错误跳转等方便编程的功能特别丰富，在程序员中被广泛使用。

简单的来说， vi 是老式的字处理器，不过功能已经很齐全了，但是还是有可以进步的地方。 vim 则可以说是程序开发者的一项很好用的工具。 连 vim 的官方网站 (http://www.vim.org) 自己也说 vim 是一个程序开发工具而不是文字处理软件。

vi/vim 的使用

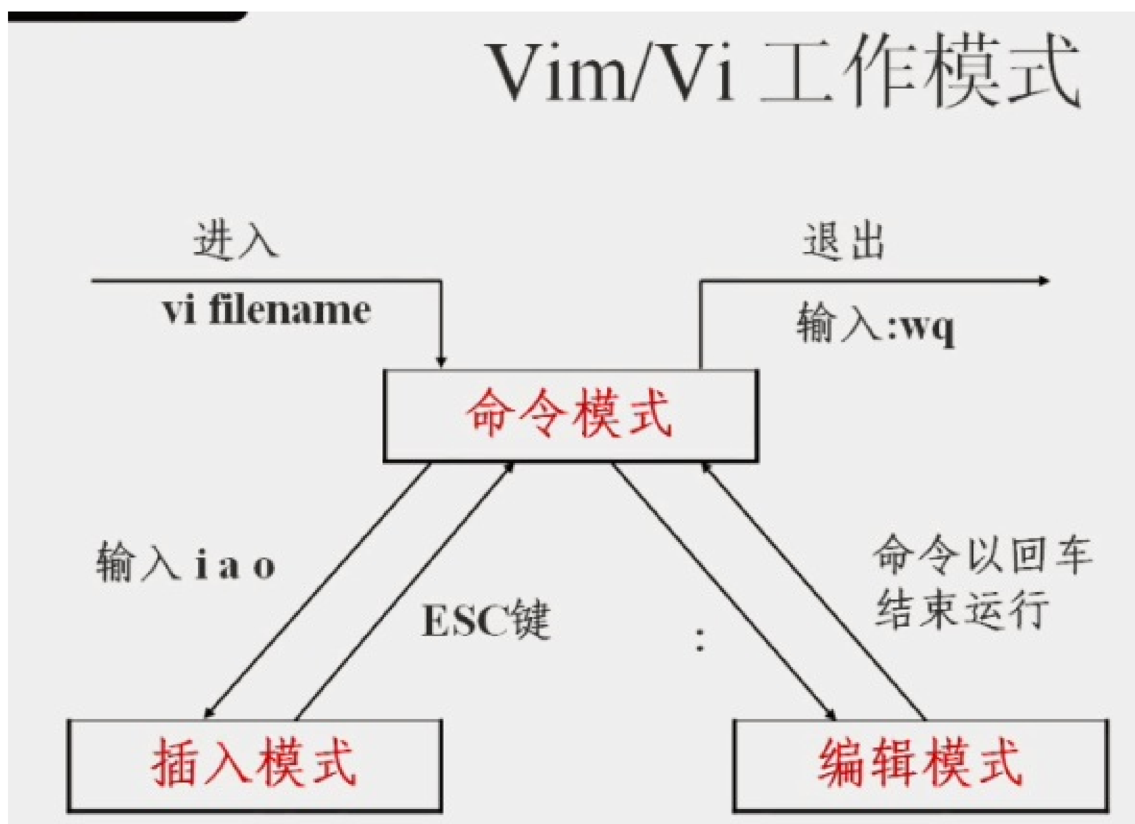
基本上 vi/vim 共分为三种模式，分别是一般模式、编辑模式与指令列命令模式。 这三种模式的作用分别是：

- 一般模式：
  以 vi 打开一个档案就直接进入一般模式了(这是默认的模式)。在这个模式中， 你可以使用『上下左右』按键来移动光标，你可以使用『删除字符』或『删除整行』来处理档案内容， 也可以使用『复制、贴上』来处理你的文件数据。

- 编辑模式：
  在一般模式中可以进行删除、复制、贴上等等的动作，但是却无法编辑文件内容的！ 要等到你按下『i, I, o, O, a, A, r, R』等任何一个字母之后才会进入编辑模式。注意了！通常在 Linux 中，按下这些按键时，在画面的左下方会出现『INSERT 或 REPLACE 』的字样，此时才可以进行编辑。而如果要回到一般模式时， 则必须要按下『Esc』这个按键即可退出编辑模式。

- 指令列命令模式：
  在一般模式当中，输入『 : / ? 』三个中的任何一个按钮，就可以将光标移动到最底下那一行。在这个模式当中， 可以提供你『搜寻资料』的动作，而读取、存盘、大量取代字符、离开 vi 、显示行号等等的动作则是在此模式中达成的！

简单的说，我们可以将这三个模式想成底下的图标来表示：



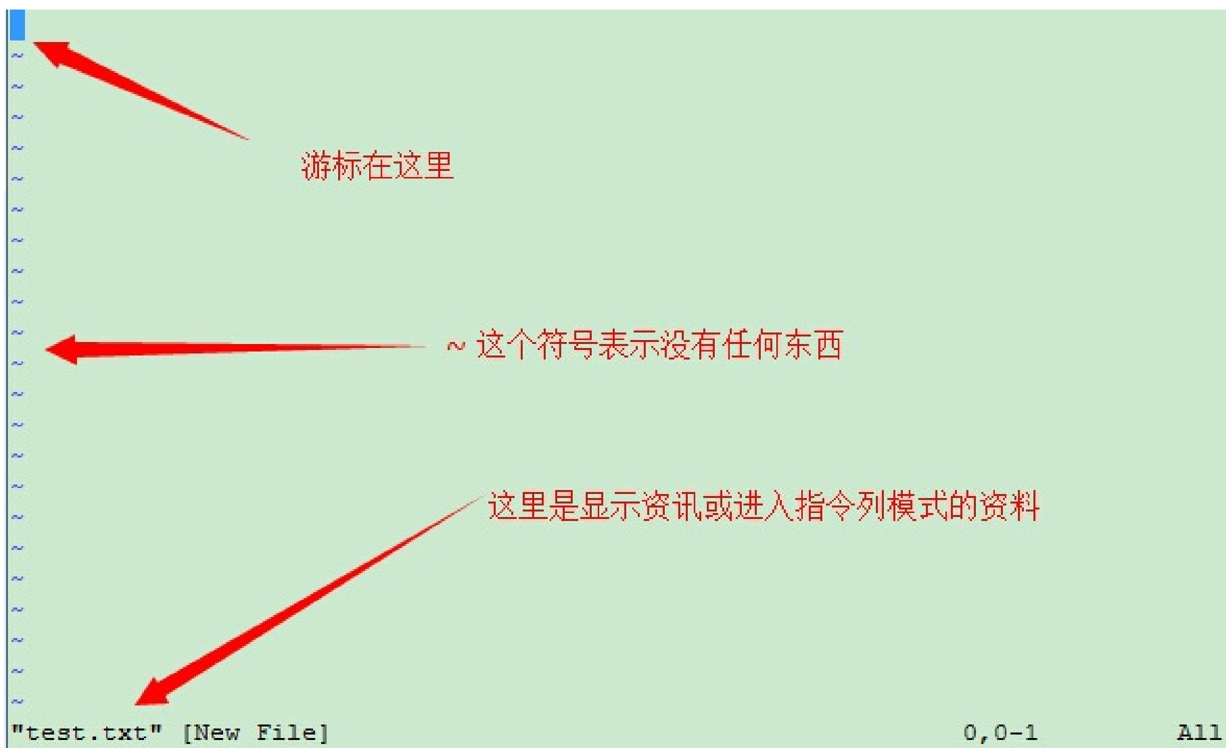
vi/vim 使用实例

使用 vi/vim 进入一般模式

如果你想要使用 vi 来建立一个名为 test.txt 的文件时，你可以这样做：

```
[root@www ~]# vi test.txt
```

直接输入 **vi** 文件名 就能够进入 vi 的一般模式了。请注意，记得 vi 后面一定要加文件名，不管该文件存在与否！



按下 i 进入编辑模式，开始编辑文字

在一般模式之中，只要按下 i, o, a 等字符就可以进入编辑模式了！

在编辑模式当中，你可以发现在左下角状态栏中会出现 –INSERT- 的字样，那就是可以输入任意字符的提示。

这个时候，键盘上除了 [Esc] 这个按键之外，其他的按键都可以视作为一般的输入按钮了，所以你可以进行任何的编辑。



按下 [ESC] 按钮回到一般模式

好了，假设我已经按照上面的样式给他编辑完毕了，那么应该要如何退出呢？是的！没错！就是给他按下 [Esc] 这个按钮即可！马上你就会发现画面左下角的 – INSERT – 不见了！

在一般模式中按下 :wq 储存后离开 vi

OK，我们要存档了，存盘并离开的指令很简单，输入『:wq』即可保存离开！



OK! 这样我们就成功创建了一个 test.txt 的文件。是不是很简单。

vi/vim 按键说明

除了上面简易范例的 i, [Esc], :wq 之外，其实 vim 还有非常多的按键可以使用。

第一部份：一般模式可用的按钮说明，光标移动、复制贴上、搜寻取代等

移动光标的方法	
h 或 向左箭头键(←)	光标向左移动一个字符
j 或 向下箭头键(↓)	光标向下移动一个字符
k 或 向上箭头键(↑)	光标向上移动一个字符
l 或 向右箭头键(→)	光标向右移动一个字符
如果你将右手放在键盘上的话，你会发现 hjkl 是排列在一起的，因此可以使用这四个按钮来移动光标。 如果想要进行	

多次移动的话，例如向下移动 30 行，可以使用 "30j" 或 "30↓" 的组合按键，亦即加上想要进行的次数(数字)后，按下动作即可！	
[Ctrl] + [f]	屏幕『向下』移动一页，相当于 [Page Down] 按键 (常用)
[Ctrl] + [b]	屏幕『向上』移动一页，相当于 [Page Up] 按键 (常用)
[Ctrl] + [d]	屏幕『向下』移动半页
[Ctrl] + [u]	屏幕『向上』移动半页
+	光标移动到非空格符的下一列
-	光标移动到非空格符的上一列
n	那个 n 表示『数字』，例如 20 。按下数字后再按空格键，光标会向右移动这一行的 n 个字符。例如 20 则光标会向后面移动 20 个字符距离。
0 或功能键[Home]	这是数字『 0 』：移动到这一行的最前面字符处 (常用)
$ 或功能键[End]	移动到这一行的最后面字符处(常用)
H	光标移动到这个屏幕的最上方那一行的第一个字符
M	光标移动到这个屏幕的中央那一行的第一个字符
L	光标移动到这个屏幕的最下方那一行的第一个字符
G	移动到这个档案的最后一行(常用)
nG	n 为数字。移动到这个档案的第 n 行。例如 20G 则会移动到这个档案的第 20 行(可配合 :set nu)
gg	移动到这个档案的第一行，相当于 1G 啊！ (常用)
n	n 为数字。光标向下移动 n 行(常用)
搜寻与取代	
/word	向光标之下寻找一个名称为 word 的字符串。例如要在档案内搜寻 vbird 这个字符串，就输入 /vbird 即可！ (常用)
?word	向光标之上寻找一个字符串名称为 word 的字符串。
	这个 n 是英文按键。代表重复前一个搜寻的

n	动作。举例来说， 如果刚刚我们执行 /vbird 去向下搜寻 vbird 这个字符串，则按下 n 后，会向下继续搜寻下一个名称为 vbird 的字符串。如果是执行 ?vbird 的话，那么按下 n 则会向上继续搜寻名称为 vbird 的字符串！
N	这个 N 是英文按键。与 n 刚好相反，为『反向』进行前一个搜寻动作。 例如 /vbird 后，按下 N 则表示『向上』搜寻 vbird 。
使用 /word 配合 n 及 N 是非常有帮助的！可以让你重复的找到一些你搜寻的关键词！	
:n1,n2s/word1/word2/g	n1 与 n2 为数字。在第 n1 与 n2 行之间寻找 word1 这个字符串，并将该字符串取代为 word2 ！举例来说，在 100 到 200 行之间搜寻 vbird 并取代为 VBIRD 则：『:100,200s/vbird/VBIRD/g』。(常用)
:1,$s/word1/word2/g	从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！(常用)
:1,$s/word1/word2/gc	从第一行到最后一行寻找 word1 字符串，并将该字符串取代为 word2 ！且在取代前显示提示字符给用户确认 (confirm) 是否需要取代！(常用)
x, X	在一行字当中，x 为向后删除一个字符 (相当于 [del] 按键)， X 为向前删除一个字符(相当于 [backspace] 亦即是退格键) (常用)
nx	n 为数字，连续向后删除 n 个字符。举例来说，我要连续删除 10 个字符， 『10x』。
dd	删除游标所在的那一整列(常用)
ndd	n 为数字。删除光标所在的向下 n 列，例如 20dd 则是删除 20 列 (常用)
d1G	删除光标所在到第一行的所有数据
dG	删除光标所在到最后一行的所有数据
d$	删除游标所在处，到该行的最后一个字符
d0	那个是数字的 0 ，删除游标所在处，到该行的最前面一个字符
yy	复制游标所在的那一行(常用)
nyy	n 为数字。复制光标所在的向下 n 列，例如 20yy 则是复制 20 列(常用)
y1G	复制游标所在列到第一列的所有数据
yG	复制游标所在列到最后一列的所有数据

y0	复制光标所在的那个字符到该行行首的所有数据
y$	复制光标所在的那个字符到该行行尾的所有数据
p, P	p 为将已复制的数据在光标下一行贴上，P 则为贴在游标上一行！ 举例来说，我目前光标在第 20 行，且已经复制了 10 行数据。则按下 p 后， 那 10 行数据会贴在原本的 20 行之后，亦即由 21 行开始贴。但如果是按下 P 呢？ 那么原本的第 20 行会被推到变成 30 行。 (常用)
J	将光标所在列与下一列的数据结合成同一列
c	重复删除多个数据，例如向下删除 10 行，[10cj]
u	复原前一个动作。(常用)
[Ctrl]+r	重做上一个动作。(常用)
这个 u 与 [Ctrl]+r 是很常用的指令！一个是复原，另一个则是重做一次～ 利用这两个功能按键，你的编辑，嘿嘿！很快乐的啦！	
.	不要怀疑！这就是小数点！意思是重复前一个动作的意思。 如果你想要重复删除、重复贴上等等动作，按下小数点『.』就好了！ (常用)

第二部份：一般模式切换到编辑模式的可用的按钮说明

进入插入或取代的编辑模式	
i, I	进入插入模式(Insert mode)： i 为『从目前光标所在处插入』， I 为『在目前所在行的第一个非空格符处开始插入』。 (常用)
a, A	进入插入模式(Insert mode)： a 为『从目前光标所在的下一个字符处开始插入』， A 为『从光标所在行的最后一个字符处开始插入』。(常用)
o, O	进入插入模式(Insert mode)： 这是英文字母 o 的大小写。o 为『在目前光标所在的下一行处插入新的一行』； O 为在目前光标所在处的上一行插入新的一行！(常用)
r, R	进入取代模式(Replace mode)：r 只会取代光标所在的那一个字符一次；R会一直取代光标所在的文字，直到按下 ESC 为止；(常用)
上面这些按键中，在 vi 画面的左下角处会出现『--INSERT--』或『--REPLACE--』的字样。 由名称就知道该动作了吧！！特别注意的是，我们上面也提过了，你想要在档案里面输入字符时， 一定要在左下角处看到 INSERT 或 REPLACE 才能输入喔！	
[Esc]	退出编辑模式，回到一般模式中(常用)

第三部份：一般模式切换到指令列模式的可用的按钮说明

指令列的储存、离开等指令	
:w	将编辑的数据写入硬盘档案中(常用)
:w!	若文件属性为『只读』时，强制写入该档案。不过，到底能不能写入， 还是跟你对该档案的档案权限有关啊！
:q	离开 vi (常用)
:q!	若曾修改过档案，又不想储存，使用！为强制离开不储存档案。
注意一下啊，那个惊叹号 (!) 在 vi 当中，常常具有『强制』的意思～	
:wq	储存后离开，若为 :wq! 则为强制储存后离开 (常用)
ZZ	这是大写的 Z 喔！若档案没有更动，则不储存离开，若档案已经被更动过，则储存后离开！
:w [filename]	将编辑的数据储存成另一个档案（类似另存新档）
:r [filename]	在编辑的数据中，读入另一个档案的数据。亦即将『filename』 这个档案内容加到游标所在行后面
:n1,n2 w [filename]	将 n1 到 n2 的内容储存成 filename 这个档案。
:! command	暂时离开 vi 到指令列模式下执行 command 的显示结果！例如 『:! ls /home』即可在 vi 当中察看 /home 底下以 ls 输出的档案信息！
vim 环境的变更	
:set nu	显示行号，设定之后，会在每一行的前缀显示该行的行号
:set nonu	与 set nu 相反，为取消行号！

特别注意，在 vi/vim 中，数字是很有意义的！数字通常代表重复做几次的意思！ 也有可能是代表去到第几个什么什么的意思。

举例来说，要删除 50 行，则是用 『50dd』 对吧！ 数字加在动作之前，如我要向下移动 20 行呢？那就是『20j』或者是『20↓』即可。

linux yum 命令

yum（ Yellow dog Updater, Modified）是一个在Fedora和RedHat以及SUSE中的Shell前端软件包管理器。

基於RPM包管理，能够从指定的服务器自动下载RPM包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软体包，无须繁琐地一次次下载、安装。

yum提供了查找、安装、删除某一个、一组甚至全部软件包的命令，而且命令简洁而又好记。

yum 语法

```
yum [options] [command] [package ...]
```

- **options**：可选，选项包括-h（帮助），-y（当安装过程提示选择全部为"yes"），-q（不显示安装的过程）等等。
- **command**：要进行的操作。
- **package**操作的对象。

yum常用命令

- 1.列出所有可更新的软件清单命令：yum check-update
- 2.更新所有软件命令：yum update
- 3.仅安装指定的软件命令：yum install <package_name>
- 4.仅更新指定的软件命令：yum update <package_name>
- 5.列出所有可安裝的软件清单命令：yum list
- 6.删除软件包命令：yum remove <package_name>
- 7.查找软件包 命令：yum search <keyword>
- 8.清除缓存命令:
  - yum clean packages: 清除缓存目录下的软件包
  - yum clean headers: 清除缓存目录下的 headers
  - yum clean oldheaders: 清除缓存目录下旧的 headers
  - yum clean, yum clean all (= yum clean packages; yum clean oldheaders) :清除缓存目录下的软件包及旧的headers

实例 1

安装 pam-devel

```
[root@www ~]# yum install pam-devel
Setting up Install Process
Parsing package install arguments
Resolving Dependencies  <==先检查软件的属性相依问题
--> Running transaction check
---> Package pam-devel.i386 0:0.99.6.2-4.el5 set to be updated
--> Processing Dependency: pam = 0.99.6.2-4.el5 for package: pam-devel
--> Running transaction check
---> Package pam.i386 0:0.99.6.2-4.el5 set to be updated
filelists.xml.gz          100% |=========================| 1.6 MB    00:05
filelists.xml.gz          100% |=========================| 138 kB    00:00
-> Finished Dependency Resolution
……(省略)
```

实例 2

移除 pam-devel

```
[root@www ~]# yum remove pam-devel
Setting up Remove Process
Resolving Dependencies  <==同样的，先解决属性相依的问题
--> Running transaction check
---> Package pam-devel.i386 0:0.99.6.2-4.el5 set to be erased
--> Finished Dependency Resolution

Dependencies Resolved

=============================================================================
 Package                 Arch       Version          Repository        Size
=============================================================================
Removing:
 pam-devel               i386       0.99.6.2-4.el5   installed         495 k

Transaction Summary
=============================================================================
Install      0 Package(s)
Update       0 Package(s)
Remove       1 Package(s)  <==还好，并没有属性相依的问题，单纯移除一个软件

Is this ok [y/N]: y
Downloading Packages:
Running rpm_check_debug
Running Transaction Test
Finished Transaction Test
Transaction Test Succeeded
Running Transaction
  Erasing    : pam-devel                    ######################### [1/1]

Removed: pam-devel.i386 0:0.99.6.2-4.el5
Complete!
```

实例 3

利用 yum 的功能，找出以 pam 为开头的软件名称有哪些？

```
[root@www ~]# yum list pam*
Installed Packages
pam.i386                  0.99.6.2-3.27.el5      installed
pam_ccreds.i386           3-5                    installed
pam_krb5.i386             2.2.14-1               installed
pam_passwdqc.i386         1.0.2-1.2.2            installed
pam_pkcs11.i386           0.5.3-23               installed
pam_smb.i386              1.1.7-7.2.1            installed
Available Packages <==底下则是『可升级』的或『未安装』的
pam.i386                  0.99.6.2-4.el5         base
pam-devel.i386            0.99.6.2-4.el5         base
pam_krb5.i386             2.2.14-10              base
```

国内 yum 源

网易（163）yum源是国内最好的yum源之一 ，无论是速度还是软件版本，都非常的不错。

将yum源设置为163 yum，可以提升软件包安装和更新的速度，同时避免一些常见软件版本无法找到。

安装步骤

首先备份/etc/yum.repos.d/CentOS-Base.repo

```
mv /etc/yum.repos.d/CentOS-Base.repo /etc/yum.repos.d/CentOS-Base.repo.backup
```

下载对应版本repo文件, 放入/etc/yum.repos.d/(操作前请做好相应备份)

- CentOS5 ：http://mirrors.163.com/.help/CentOS5-Base-163.repo
- CentOS6 ：http://mirrors.163.com/.help/CentOS6-Base-163.repo

运行以下命令生成缓存

```
yum clean all
yum makecache
```

除了网易之外，国内还有其他不错的yum源，比如中科大和搜狐。

中科大的yum源，安装方法查看：https://lug.ustc.edu.cn/wiki/mirrors/help/centos

sohu的yum源安装方法查看: http://mirrors.sohu.com/help/centos.html

Shell 编程

Shell 教程

Shell 是一个用C语言编写的程序，它是用户使用Linux的桥梁。Shell既是一种命令语言，又是一种程序设计语言。

Shell 是指一种应用程序，这个应用程序提供了一个界面，用户通过这个界面访问操作系统内核的服务。

Ken Thompson的sh是第一种Unix Shell，Windows Explorer是一个典型的图形界面Shell。

Shell 脚本

Shell 脚本（shell script），是一种为shell编写的脚本程序。

业界所说的shell通常都是指shell脚本，但读者朋友要知道，shell和shell script是两个不同的概念。

由于习惯的原因，简洁起见，本文出现的"shell编程"都是指shell脚本编程，不是指开发shell自身。

Shell 环境

Shell 编程跟java、php编程一样，只要有一个能编写代码的文本编辑器和一个能解释执行的脚本解释器就可以了。

Linux的Shell种类众多，常见的有：

- Bourne Shell（/usr/bin/sh或/bin/sh）
- Bourne Again Shell（/bin/bash）
- C Shell（/usr/bin/csh）
- K Shell（/usr/bin/ksh）
- Shell for Root（/sbin/sh）
- ……

本教程关注的是 Bash，也就是 Bourne Again Shell，由于易用和免费，Bash在日常工作中被广泛使用。同时，Bash也是大多数Linux系统默认的Shell。

在一般情况下，人们并不区分 Bourne Shell 和 Bourne Again Shell，所以，像 **#!/bin/sh**，它同样也可以改为**#!/bin/bash**。

`#!` 告诉系统其后路径所指定的程序即是解释此脚本文件的Shell程序。

第一个shell脚本

打开文本编辑器(可以使用vi/vim命令来创建文件)，新建一个文件test.sh，扩展名为sh（sh代表shell），扩展名并不影响脚本执行，见名知意就好，如果你用php写shell 脚本，扩展名就用php好了。

输入一些代码，第一行一般是这样：

```
#!/bin/bash
echo "Hello World !"
```

" `#!` " 是一个约定的标记，它告诉系统这个脚本需要什么解释器来执行，即使用哪一种Shell。

echo命令用于向窗口输出文本。

运行Shell脚本有两种方法：

1、作为可执行程序

将上面的代码保存为test.sh，并cd到相应目录：

```
chmod +x ./test.sh  #使脚本具有执行权限
./test.sh  #执行脚本
```

注意，一定要写成./test.sh，而不是test.sh，运行其它二进制的程序也一样，直接写test.sh，linux系统会去PATH里寻找有没有叫test.sh的，而只有/bin, /sbin, /usr/bin，/usr/sbin等在PATH里，你的当前目录通常不在PATH里，所以写成test.sh是会找不到命令的，要用./test.sh告诉系统说，就在当前目录找。

2、作为解释器参数

这种运行方式是，直接运行解释器，其参数就是shell脚本的文件名，如：

```
/bin/sh test.sh
/bin/php test.php
```

这种方式运行的脚本，不需要在第一行指定解释器信息，写了也没用。

Shell 变量

定义变量时，变量名不加美元符号（$，PHP语言中变量需要），如：

```
your_name="w3cschool.cc"
```

注意，变量名和等号之间不能有空格，这可能和你熟悉的所有编程语言都不一样。同时，变量名的命名须遵循如下规则：

- 首个字符必须为字母（a-z，A-Z）。
- 中间不能有空格，可以使用下划线（_）。
- 不能使用标点符号。
- 不能使用bash里的关键字（可用help命令查看保留关键字）。

除了显式地直接赋值，还可以用语句给变量赋值，如：

```
for file in `ls /etc`
```

以上语句将 /etc 下目录的文件名循环出来。

使用变量

使用一个定义过的变量，只要在变量名前面加美元符号即可，如：

```
your_name="qinjx"
echo $your_name
echo ${your_name}
```

变量名外面的花括号是可选的，加不加都行，加花括号是为了帮助解释器识别变量的边界，比如下面这种情况：

```
for skill in Ada Coffe Action Java do
    echo "I am good at ${skill}Script"
done
```

如果不给skill变量加花括号，写成echo "I am good at $skillScript"，解释器就会把$skillScript当成一个变量（其值为空），代码执行结果就不是我们期望的样子了。

推荐给所有变量加上花括号，这是个好的编程习惯。

已定义的变量，可以被重新定义，如：

```
your_name="tom"
echo $your_name
your_name="alibaba"
echo $your_name
```

这样写是合法的，但注意，第二次赋值的时候不能写$your_name="alibaba"，使用变量的时候才加美元符（$）。

Shell 字符串

字符串是shell编程中最常用最有用的数据类型（除了数字和字符串，也没啥其它类型好用了），字符串可以用单引号，也可以用双引号，也可以不用引号。单双引号的区别跟PHP类似。

单引号

```
str='this is a string'
```

单引号字符串的限制：

- 单引号里的任何字符都会原样输出，单引号字符串中的变量是无效的；
- 单引号字串中不能出现单引号（对单引号使用转义符后也不行）。

双引号

```
your_name='qinjx'
str="Hello, I know your are \"$your_name\"! \n"
```

双引号的优点：

- 双引号里可以有变量
- 双引号里可以出现转义字符

拼接字符串

```
your_name="qinjx"
greeting="hello, "$your_name" !"
greeting_1="hello, ${your_name} !"
echo $greeting $greeting_1
```

获取字符串长度

```
string="abcd"
echo ${#string} #输出 4
```

提取子字符串

```
string="alibaba is a great company"
echo ${string:1:4} #输出liba
```

查找子字符串

```
string="alibaba is a great company"
echo `expr index "$string" is`
```

Shell 数组

bash支持一维数组（不支持多维数组），并且没有限定数组的大小。

类似与C语言，数组元素的下标由0开始编号。获取数组中的元素要利用下标，下标可以是整数或算术表达式，其值应大于或等于0。

定义数组

在Shell中，用括号来表示数组，数组元素用"空格"符号分割开。定义数组的一般形式为：

```
    数组名=(值1 值2 ... 值n)
```

例如：

```
array_name=(value0 value1 value2 value3)
```

或者

```
array_name=(
value0
value1
value2
value3
)
```

还可以单独定义数组的各个分量：

```
array_name[0]=value0
array_name[1]=value1
array_name[n]=valuen
```

可以不使用连续的下标，而且下标的范围没有限制。

读取数组

读取数组元素值的一般格式是：

```
    ${数组名[下标]}
```

例如：

```
valuen=${array_name[n]}
```

使用@符号可以获取数组中的所有元素，例如：

```
echo ${array_name[@]}
```

获取数组的长度

获取数组长度的方法与获取字符串长度的方法相同，例如：

```
# 取得数组元素的个数
length=${#array_name[@]}
# 或者
length=${#array_name[*]}
# 取得数组单个元素的长度
lengthn=${#array_name[n]}
```

Shell 注释

以"#"开头的行就是注释，会被解释器忽略。

sh里没有多行注释，只能每一行加一个#号。只能像这样：

```
#--------------------------------------------
# 这是一个自动打ipa的脚本，基于webfrogs的ipa-build书写：
# https://github.com/webfrogs/xcode_shell/blob/master/ipa-build
# 功能：自动为etao ios app打包，产出物为14个渠道的ipa包
# 特色：全自动打包，不需要输入任何参数
#--------------------------------------------
##### 用户配置区 开始 #####
#
#
# 项目根目录，推荐将此脚本放在项目的根目录，这里就不用改了
# 应用名，确保和Xcode里Product下的target_name.app名字一致
#
##### 用户配置区 结束  #####
```

如果在开发过程中，遇到大段的代码需要临时注释起来，过一会儿又取消注释，怎么办呢？

每一行加个#符号太费力了，可以把这一段要注释的代码用一对花括号括起来，定义成一个函数，没有地方调用这个函数，这块代码就不会执行，达到了和注释一样的效果。

Shell echo命令

Shell 的 echo 指令与 PHP 的 echo 指令类似，都是用于字符串的输出。命令格式：

```
echo string
```

您可以使用echo实现更复杂的输出格式控制。

1.显示普通字符串:

```
echo "It is a test"
```

这里的双引号完全可以省略，以下命令与上面实例效果一致：

```
echo It is a test
```

2.显示转义字符

```
echo "\"It is a test\""
```

结果将是:

```
"It is a test"
```

同样，双引号也可以省略

3.显示变量

read 命令从标准输入中读取一行,并把输入行的每个字段的值指定给 shell 变量

```
#!/bin/sh
read name
echo "$name It is a test"
```

以上代码保存为 test.sh，name 接收标准输入的变量，结果将是:

```
[root@www ~]# sh test.sh
OK                     #标准输入
OK It is a test        #输出
```

4.显示换行

```
echo -e "OK!\n" # -e 开启转义
echo "It it a test"
```

输出结果：

```
OK!

It it a test
```

5.显示不换行

```
#!/bin/sh
echo -e "OK! \c" # -e 开启转义 \c 不换行
echo "It is a test"
```

输出结果：

```
OK! It is a test
```

6.显示结果定向至文件

```
echo "It is a test" > myfile
```

7.原样输出字符串，不进行转义或取变量(用单引号)

```
echo '$name\"'
```

输出结果：

```
$name\"
```

8.显示命令执行结果

```
echo `date`
```

结果将显示当前日期

```
Thu Jul 24 10:08:46 CST 2014
```

Shell test命令

Shell中的 test 命令用于检查某个条件是否成立，它可以进行数值、字符和文件三个方面的测试。

数值测试

参数	说明
-eq	等于则为真
-ne	不等于则为真
-gt	大于则为真
-ge	大于等于则为真
-lt	小于则为真
-le	小于等于则为真

实例演示：

```
num1=100
num2=100
if test $[num1] -eq $[num2]
then
    echo 'The two numbers are equal!'
else
    echo 'The two numbers are not equal!'
fi
```

输出结果：

```
The two numbers are equal!
```

字符串测试

参数	说明
=	等于则为真
!=	不相等则为真
-z 字符串	字符串长度伪则为真
-n 字符串	字符串长度不伪则为真

实例演示：

```
num1=100
num2=100
if test num1=num2
then
    echo 'The two strings are equal!'
else
    echo 'The two strings are not equal!'
fi
```

输出结果：

```
The two strings are equal!
```

文件测试

参数	说明
-e 文件名	如果文件存在则为真
-r 文件名	如果文件存在且可读则为真
-w 文件名	如果文件存在且可写则为真
-x 文件名	如果文件存在且可执行则为真
-s 文件名	如果文件存在且至少有一个字符则为真
-d 文件名	如果文件存在且为目录则为真
-f 文件名	如果文件存在且为普通文件则为真
-c 文件名	如果文件存在且为字符型特殊文件则为真
-b 文件名	如果文件存在且为块特殊文件则为真

实例演示：

```
cd /bin
if test -e ./bash
then
    echo 'The file already exists!'
else
    echo 'The file does not exists!'
fi
```

输出结果：

```
The file already exists!
```

另外，Shell还提供了与(!)、或(-o)、非(-a)三个逻辑操作符用于将测试条件连接起来，其优先级为："!"最高，"-a"次之，"-o"最低。例如：

```
cd /bin
if test -e ./notFile -o ./bash
then
    echo 'One file exists at least!'
else
    echo 'Both dose not exists!'
fi
```

输出结果：

```
One file exists at least!
```

Shell 流程控制

和Java、PHP等语言不一样，sh的流程控制不可为空，如(以下为PHP流程控制写法)：

```
<?php
if (isset($_GET["q"])) {
    search(q);
}
else {
    //do nothing
}
```

在sh/bash里可不能这么写，如果else分支没有语句执行，就不要写这个else，就像这样

if else

if

if 语句语法格式：

```
if condition
then
    command1
    command2
    ...
    commandN
fi
```

写成一行（适用于终端命令提示符）：

```
if `ps -ef | grep ssh`;  then echo hello; fi
```

末尾的fi就是if倒过来拼写，后面还会遇到类似的。

if else

if else 语法格式：

```
if condition
then
    command1
    command2
    ...
    commandN
else
    command
fi
```

if else-if else

if else-if else 语法格式：

```
if condition1
then
    command1
elif condition2
    command2
else
    commandN
fi
```

if else语句经常与test命令结合使用，如下所示：

```
num1=$[2*3]
num2=$[1+5]
if test $[num1] -eq $[num2]
then
    echo 'The two numbers are equal!'
else
    echo 'The two numbers are not equal!'
fi
```

输出结果：

```
The two numbers are equal!
```

for 循环

与其他编程语言类似，Shell支持for循环。

for循环一般格式为：

```
for var in item1 item2 ... itemN
do
    command1
    command2
    ...
    commandN
done
```

写成一行：

```
for var in item1 item2 ... itemN; do command1; command2… done;
```

当变量值在列表里，for循环即执行一次所有命令，使用变量名获取列表中的当前取值。命令可为任何有效的shell命令和语句。in列表可以包含替换、字符串和文件名。

in列表是可选的，如果不用它，for循环使用命令行的位置参数。

例如，顺序输出当前列表中的数字：

```
for loop in 1 2 3 4 5
do
    echo "The value is: $loop"
done
```

输出结果：

```
The value is: 1
The value is: 2
The value is: 3
The value is: 4
The value is: 5
```

顺序输出字符串中的字符：

```
for str in 'This is a string'
do
    echo $str
done
```

输出结果：

```
This is a string
```

while 语句

while循环用于不断执行一系列命令，也用于从输入文件中读取数据；命令通常为测试条件。其格式为：

```
while condition
do
    command
done
```

命令执行完毕，控制返回循环顶部，从头开始直至测试条件为假。

以下是一个基本的while循环，测试条件是：如果COUNTER小于5，那么条件返回真。COUNTER从0开始，每次循环处理时，COUNTER加1。运行上述脚本，返回数字1到5，然后终止。

```
COUNTER=0
while [ $COUNTER -lt 5 ]
do
    COUNTER='expr $COUNTER+1'
    echo $COUNTER
done
```

运行脚本，输出：

```
1
2
3
4
5
```

while循环可用于读取键盘信息。下面的例子中，输入信息被设置为变量FILM，按<Ctrl-D>结束循环。

```
echo 'type <CTRL-D> to terminate'
echo -n 'enter your most liked film: ''
while read FILM
do
    echo "Yeah! great film the $FILM"
done
```

运行脚本，输出类似下面：

```
type <CTRL-D> to terminate
enter your most liked film: Sound of Music
Yeah! great film the Sound of Music
```

无限循环

无限循环语法格式：

```
while :
do
    command
done
```

或者

```
while true
do
    command
done
```

或者

```
for (( ; ; ))
```

until 循环

until循环执行一系列命令直至条件为真时停止。

until循环与while循环在处理方式上刚好相反。

一般while循环优于until循环，但在某些时候—也只是极少数情况下，until循环更加有用。

until 语法格式:

```
until condition
do
    command
done
```

条件可为任意测试条件，测试发生在循环末尾，因此循环至少执行一次—请注意这一点。

case

Shell case语句为多选择语句。可以用case语句匹配一个值与一个模式，如果匹配成功，执行相匹配的命令。case语句格式如下：

```
case 值 in
模式1)
    command1
    command2
    ...
    commandN
    ;;
模式2)
    command1
    command2
    ...
    commandN
    ;;
esac
```

case工作方式如上所示。取值后面必须为单词in，每一模式必须以右括号结束。取值可以为变量或常数。匹配发现取值符合某一模式后，其间所有命令开始执行直至 ;;。

取值将检测匹配的每一个模式。一旦模式匹配，则执行完匹配模式相应命令后不再继续其他模式。如果无一匹配模式，使用星号 * 捕获该值，再执行后面的命令。

下面的脚本提示输入1到4，与每一种模式进行匹配：

```
echo 'Input a number between 1 to 4'
echo 'Your number is:\c'
read aNum
case $aNum in
    1)  echo 'You select 1'
    ;;
    2)  echo 'You select 2'
    ;;
    3)  echo 'You select 3'
    ;;
    4)  echo 'You select 4'
    ;;
    *)  echo 'You do not select a number between 1 to 4'
    ;;
esac
```

输入不同的内容，会有不同的结果，例如：

```
Input a number between 1 to 4
Your number is:3
You select 3
```

跳出循环

在循环过程中，有时候需要在未达到循环结束条件时强制跳出循环，Shell使用两个命令来实现该功能：break和continue。

break命令

break命令允许跳出所有循环（终止执行后面的所有循环）。

下面的例子中，脚本进入死循环直至用户输入数字大于5。要跳出这个循环，返回到shell提示符下，需要使用break命令。

```
#!/bin/bash
while :
do
    echo -n "Input a number between 1 to 5: "
    read aNum
    case $aNum in
        1|2|3|4|5) echo "Your number is $aNum!"
        ;;
        *) echo "You do not select a number between 1 to 5, game is over!"
            break
        ;;
    esac
done
```

continue

```
continue命令与break命令类似，只有一点差别，它不会跳出所有循环，仅仅跳出当前循环。
```

对上面的例子进行修改：

```
#!/bin/bash
while :
do
    echo -n "Input a number between 1 to 5: "
    read aNum
    case $aNum in
        1|2|3|4|5) echo "Your number is $aNum!"
        ;;
        *) echo "You do not select a number between 1 to 5!"
            continue
            echo "Game is over!"
        ;;
    esac
done
```

运行代码发现，当输入大于5的数字时，该例中的循环不会结束，语句 **echo "Game is over!"** 永远不会被执行。

esac

case的语法和C family语言差别很大，它需要一个esac（就是case反过来）作为结束标记，每个case分支用右圆括号，用两个分号表示break。

Shell 函数

linux shell 可以用户定义函数，然后在shell脚本中可以随便调用。

shell中函数的定义格式如下：

```
[ function ] funname [()]

{

    action;

    [return int;]

}
```

说明：

- 1、可以带function fun() 定义，也可以直接fun() 定义,不带任何参数。
- 2、参数返回，可以显示加：return 返回，如果不加，将以最后一条命令运行结果，作为返回值。 return后跟数值n(0-255

下面的例子定义了一个函数并进行调用：

```
#!/bin/bash
demoFun(){
    echo "This is your first shell function!"
}
echo "Function begin..."
hello
echo "Function end!"
```

输出结果：

```
Function begin...
This is your first shell function!
Function end!
```

下面定义一个带有return语句的函数：

```
#!/bin/bash
funWithReturn(){
    echo "The function is to get the sum of two numbers..."
    echo -n "Input first number: "
    read aNum
    echo -n "Input another number: "
    read anotherNum
    echo "The two numbers are $aNum and $anotherNum !"
    return $(($aNum+$anotherNum))
}
funWithReturn
echo "The sum of two numbers is $? !"
```

输出类似下面：

```
The function is to get the sum of two numbers...
Input first number: 25
Input another number: 50
The two numbers are 25 and 50 !
The sum of two numbers is 75 !
```

函数返回值在调用该函数后通过 $? 来获得。

注意：所有函数在使用前必须定义。这意味着必须将函数放在脚本开始部分，直至shell解释器首次发现它时，才可以使用。调用函数仅使用其函数名即可。

函数参数

在Shell中，调用函数时可以向其传递参数。在函数体内部，通过 $n 的形式来获取参数的值，例如，$1表示第一个参数，$2表示第二个参数...

带参数的函数示例：

```
#!/bin/bash
funWithParam(){
    echo "The value of the first parameter is $1 !"
    echo "The value of the second parameter is $2 !"
    echo "The value of the tenth parameter is $10 !"
    echo "The value of the tenth parameter is ${10} !"
    echo "The value of the eleventh parameter is ${11} !"
    echo "The amount of the parameters is $# !"
    echo "The string of the parameters is $* !"
}
funWithParam 1 2 3 4 5 6 7 8 9 34 73
```

输出结果：

```
The value of the first parameter is 1 !
The value of the second parameter is 2 !
The value of the tenth parameter is 10 !
The value of the tenth parameter is 34 !
The value of the eleventh parameter is 73 !
The amount of the parameters is 12 !
The string of the parameters is 1 2 3 4 5 6 7 8 9 34 73 !"
```

注意，$10 不能获取第十个参数，获取第十个参数需要${10}。当n>=10时，需要使用${n}来获取参数。

另外，还有几个特殊字符用来处理参数：

参数处理	说明
$#	传递到脚本的参数个数
$*	以一个单字符串显示所有向脚本传递的参数
$	脚本运行的当前进程ID号
$!	后台运行的最后一个进程的ID号
$@	与$#相同，但是使用时加引号，并在引号中返回每个参数。
$-	显示Shell使用的当前选项，与set命令功能相同。
$?	显示最后命令的退出状态。0表示没有错误，其他任何值表明有错误。

Linux命令大全

Linux命令大全 - 文件管理

cat	chattr	chgrp	chmod
chown	cksum	cmp	diff
diffstat	file	find	git
gitview	indent	cut	ln
less	locate	lsattr	mattrib
mc	mdel	mdir	mktemp
more	mmove	mread	mren
mtools	mtoolstest	mv	od
paste	patch	rcp	rm
slocate	split	tee	tmpwatch
touch	umask	which	cp
whereis	mcopy	mshowfat	rhmask
scp	awk		

Linux cat命令

命令：cat

cat命令用于把档案串连接后传到基本输出（萤幕或加 > fileName 到另一个档案）

使用权限

所有使用者

语法格式

```
cat [-AbeEnstTuv] [--help] [--version] fileName
```

参数说明：

-n 或 --number 由 1 开始对所有输出的行数编号

-b 或 --number-nonblank 和 -n 相似，只不过对于空白行不编号

-s 或 --squeeze-blank 当遇到有连续两行以上的空白行，就代换为一行的空白行

-v 或 --show-nonprinting

实例：

把 textfile1 的档案内容加上行号后输入 textfile2 这个档案里

```
cat -n textfile1 &gt; textfile2
```

把 textfile1 和 textfile2 的档案内容加上行号（空白行不加）之后将内容附加到 textfile3 里。

```
cat -b textfile1 textfile2 &gt;&gt; textfile3
```

清空/etc/test.txt档案内容

```
cat /dev/null &gt; /etc/test.txt
```

cat 也可以用来制作镜像文件。例如要制作软碟的像文件，将软碟放好后打

```
cat /dev/fd0 &gt; OUTFILE
```

相反的，如果想把 image file 写到软碟，请打

```
cat IMG_FILE &gt; /dev/fd0
```

注：

- 1. OUTFILE 指输出的镜像文件名。
- 2. IMG_FILE 指镜像文件。
- 3. 若从镜像文件写回 device 时，device 容量需与相当。
- 4. 通常用在制作开机磁片。

Linux chattr命令

Linux chattr命令用于改变文件属性。

这项指令可改变存放在ext2文件系统上的文件或目录属性，这些属性共有以下8种模式：

1. a：让文件或目录仅供附加用途。
2. b：不更新文件或目录的最后存取时间。
3. c：将文件或目录压缩后存放。
4. d：将文件或目录排除在倾倒操作之外。
5. i：不得任意更动文件或目录。
6. s：保密性删除文件或目录。
7. S：即时更新文件或目录。
8. u：预防以外删除。

语法

```
chattr [-RV][-v<版本编号>][+/-/=<属性>][文件或目录...]
```

参数

-R 递归处理，将指定目录下的所有文件及子目录一并处理。

-v<版本编号> 设置文件或目录版本。

-V 显示指令执行过程。

+<属性> 开启文件或目录的该项属性。

-<属性> 关闭文件或目录的该项属性。

=<属性> 指定文件或目录的该项属性。

实例

用chattr命令防止系统中某个关键文件被修改：

```
chattr +i /etc/resolv.conf
```

```
lsattr /etc/resolv.conf
```

会显示如下属性

```
----i-------- /etc/resolv.conf
```

让某个文件只能往里面追加数据，但不能删除，适用于各种日志文件：

```
chattr +a /var/log/messages
```

Linux chgrp命令

Linux chgrp命令用于变更文件或目录的所属群组。

在UNIX系统家族里，文件或目录权限的掌控以拥有者及所属群组来管理。您可以使用chgrp指令去变更文件与目录的所属群组，设置方式采用群组名称或群组识别码皆可。

语法

```
chgrp [-cfhRv][--help][--version][所属群组][文件或目录...] 或 chgrp [-cfhRv][--help][--refer
```

参数说明

-c或--changes 效果类似"-v"参数，但仅回报更改的部分。

-f或--quiet或--silent 　不显示错误信息。

-h或--no-dereference 　只对符号连接的文件作修改，而不更动其他任何相关文件。

-R或--recursive 　递归处理，将指定目录下的所有文件及子目录一并处理。

-v或--verbose 　显示指令执行过程。

--help 　在线帮助。

--reference=<参考文件或目录> 　把指定文件或目录的所属群组全部设成和参考文件或目录的所属群组相同。

--version 　显示版本信息。

实例

实例1：改变文件的群组属性：

```
chgrp -v bin log2012.log
```

输出：

```
[root@localhost test]# ll
---xrw-r-- 1 root root 302108 11-13 06:03 log2012.log
[root@localhost test]# chgrp -v bin log2012.log
```

"log2012.log" 的所属组已更改为 bin

```
[root@localhost test]# ll
---xrw-r-- 1 root bin  302108 11-13 06:03 log2012.log
```

说明： 将log2012.log文件由root群组改为bin群组

实例2：根据指定文件改变文件的群组属性

```
chgrp --reference=log2012.log log2013.log
```

输出：

```
[root@localhost test]# ll
---xrw-r-- 1 root bin  302108 11-13 06:03 log2012.log
-rw-r--r-- 1 root root     61 11-13 06:03 log2013.log
[root@localhost test]#  chgrp --reference=log2012.log log2013.log
[root@localhost test]# ll
---xrw-r-- 1 root bin  302108 11-13 06:03 log2012.log
-rw-r--r-- 1 root bin      61 11-13 06:03 log2013.log
```

说明： 改变文件log2013.log 的群组属性，使得文件log2013.log的群组属性和参考文件log2012.log的群组属性相同

Linux chmod命令

Linux/Unix 的文件调用权限分为三级 : 文件拥有者、群组、其他。利用 chmod 可以藉以控制文件如何被他人所调用。

使用权限 : 所有使用者

语法

```
chmod [-cfvR] [--help] [--version] mode file...
```

参数说明

- mode : 权限设定字串，格式如下 :

  ```
  [ugoa...][[+-=][rwxX]...][,...]
  ```

  其中 :

  - u 表示该文件的拥有者，g 表示与该文件的拥有者属于同一个群体(group)者，o 表示其他以外的人，a 表示这三者皆是。
  -
    - 表示增加权限、- 表示取消权限、= 表示唯一设定权限。
  - r 表示可读取，w 表示可写入，x 表示可执行，X 表示只有当该文件是个子目录或者该文件已经被设定过为可执行。 *-c : 若该文件权限确实已经更改，才显示其更改动作* -f : 若该文件权限无法被更改也不要显示错误讯息 *-v : 显示权限变更的详细资料* -R : 对目前目录下的所有文件与子目录进行相同的权限变更(即以递回的方式逐个变更) *--help : 显示辅助说明* --version : 显示版本

实例

将文件 file1.txt 设为所有人皆可读取 :

```
chmod ugo+r file1.txt
```

将文件 file1.txt 设为所有人皆可读取 :

```
chmod a+r file1.txt
```

将文件 file1.txt 与 file2.txt 设为该文件拥有者，与其所属同一个群体者可写入，但其他以外的人则不可写入：

```
chmod ug+w,o-w file1.txt file2.txt
```

将 ex1.py 设定为只有该文件拥有者可以执行：

```
chmod u+x ex1.py
```

将目前目录下的所有文件与子目录皆设为任何人可读取：

```
chmod -R a+r *
```

此外chmod也可以用数字来表示权限如：

```
chmod 777 file
```

语法为：

```
chmod abc file
```

其中a,b,c各为一个数字，分别表示User、Group、及Other的权限。

r=4，w=2，x=1

- 若要rwx属性则4+2+1=7；
- 若要rw-属性则4+2=6；
- 若要r-x属性则4+1=5。

```
chmod a=rwx file
```

和

```
chmod 777 file
```

效果相同

```
chmod ug=rwx,o=x file
```

和

```
chmod 771 file
```

效果相同

若用chmod 4755 filename可使此程序具有root的权限

Linux chown命令

Linux/Unix 是多人多工操作系统，所有的文件皆有拥有者。利用 chown 将指定文件的拥有者改为指定的用户或组，用户可以是用户名或者用户ID；组可以是组名或者组ID；文件是以空格分开的要改变权限的文件列表，支持通配符。 。

一般来说，这个指令只有是由系统管理者(root)所使用，一般使用者没有权限可以改变别人的文件拥有者，也没有权限可以自己的文件拥有者改设为别人。只有系统管理者(root)才有这样的权限。

使用权限 : root

语法

```
chmod [-cfhvR] [--help] [--version] user[:group] file...
```

参数 :

- user : 新的文件拥有者的使用者 ID
- group : 新的文件拥有者的使用者群体(group)
- -c : 若该文件拥有者确实已经更改，才显示其更改动作
- -f : 若该文件拥有者无法被更改也不要显示错误讯息
- -h : 只对于连结(link)进行变更，而非该 link 真正指向的文件
- -v : 显示拥有者变更的详细资料
- -R : 对目前目录下的所有文件与子目录进行相同的拥有者变更(即以递回的方式逐个变更)
- --help : 显示辅助说明
- --version : 显示版本

实例

将文件 file1.txt 的拥有者设为 users 群体的使用者 jessie :

```
chown jessie:users file1.txt
```

将目前目录下的所有文件与子目录的拥有者皆设为 users 群体的使用者 lamport :

```
chmod -R lamport:users *
```

Linux cksum命令

Linux cksum命令用于检查文件的CRC是否正确。确保文件从一个系统传输到另一个系统的过程中不被损坏。

CRC是一种排错检查方式，该校验法的标准由CCITT所指定，至少可检测到99.998%的已知错误。

指定文件交由指令"cksum"进行校验后，该指令会返回校验结果供用户核对文件是否正确无误。若不指定任何文件名称或是所给予的文件名为"-"，则指令"cksum"会从标准输入设备中读取数据。

语法

```
cksum [--help][--version][文件...]
```

参数：

- --help：在线帮助。
- --version：显示版本信息。
- 文件...:需要进行检查的文件路径

实例

使用指令"cksum"计算文件"testfile1"的完整性，输入如下命令：

```
$ cksum testfile1
```

以上命令执行后，将输出校验码等相关的信息，具体输出信息如下所示：

```
1263453430 78 testfile1         //输出信息
```

上面的输出信息中，"1263453430"表示校验码，"78"表示字节数。

注意：如果文件中有任何字符被修改，都将改变计算后CRC校验码的值。

Linux cmp命令

Linux cmp命令用于比较两个文件是否有差异。

当相互比较的两个文件完全一样时，则该指令不会显示任何信息。若发现有所差异，预设会标示出第一个不同之处的字符和列数编号。若不指定任何文件名称或是所给予的文件名为"-"，则cmp指令会从标准输入设备读取数据。

语法

```
cmp [-clsv][-i <字符数目>][--help][第一个文件][第二个文件]
```

参数：

- -c或--print-chars 　除了标明差异处的十进制字码之外，一并显示该字符所对应字符。
- -i<字符数目>或--ignore-initial=<字符数目> 　指定一个数目。
- -l或--verbose 　标示出所有不一样的地方。
- -s或--quiet或--silent 　不显示错误信息。
- -v或--version 　显示版本信息。
- --help 　在线帮助。

实例

要确定两个文件是否相同，请输入：

```
cmp prog.o.bak prog.o
```

这比较 prog.o.bak 和 prog.o。如果文件相同，则不显示消息。如果文件不同，则显示第一个不同的位置；例如：

```
prog.o.bak prog.o differ: char 4, line 1
```

如果显示消息 cmp: EOF on prog.o.bak，则 prog.o 的第一部分与 prog.o.bak 相同，但在 prog.o 中还有其他数据。

Linux diff命令

Linux diff命令用于比较文件的差异。

diff以逐行的方式，比较文本文件的异同处。所是指定要比较目录，则diff会比较目录中相同文件名的文件，但不会比较其中子目录。

语法

```
diff [-abBcdefHilnNpPqrstTuvwy][-<行数>][-C <行数>][-D <巨集名称>][-I <字符或字符串>][-S <文件
```

参数：

- -<行数>　指定要显示多少行的文本。此参数必须与-c或-u参数一并使用。 *-a*或*--text diff*预设只会逐行比较文本文件。 -b或--ignore-space-change　不检查空格字符的不同。
- -B或--ignore-blank-lines　不检查空白行。
- -c　显示全部内文，并标出不同之处。
- -C<行数>或--context<行数>　与执行"-c-<行数>"指令相同。
- -d或--minimal　使用不同的演算法，以较小的单位来做比较。
- -D<巨集名称>或ifdef<巨集名称>　此参数的输出格式可用于前置处理器巨集。
- -e或--ed　此参数的输出格式可用于ed的script文件。
- -f或-forward-ed　输出的格式类似ed的script文件，但按照原来文件的顺序来显示不同处。
- -H或--speed-large-files　比较大文件时，可加快速度。
- -I<字符或字符串>或--ignore-matching-lines<字符或字符串>　若两个文件在某几行有所不同，而这几行同时都包含了选项中指定的字符或字符串，则不显示这两个文件的差异。
- -i或--ignore-case　不检查大小写的不同。
- -l或--paginate　将结果交由pr程序来分页。
- -n或--rcs　将比较结果以RCS的格式来显示。
- -N或--new-file　在比较目录时，若文件A仅出现在某个目录中，预设会显示：
- Only in目录：文件A若使用-N参数，则diff会将文件A与一个空白的文件比较。
- -p　若比较的文件为C语言的程序码文件时，显示差异所在的函数名称。
- -P或--unidirectional-new-file　与-N类似，但只有当第二个目录包含了一个第一个目录所没有的文件时，才会将这个文件与空白的文件做比较。
- -q或--brief　仅显示有无差异，不显示详细的信息。
- -r或--recursive　比较子目录中的文件。

- -s或--report-identical-files　若没有发现任何差异，仍然显示信息。
- -S<文件>或--starting-file<文件>　在比较目录时，从指定的文件开始比较。
- -t或--expand-tabs　在输出时，将tab字符展开。
- -T或--initial-tab　在每行前面加上tab字符以便对齐。
- -u,-U<列数>或--unified=<列数>　以合并的方式来显示文件内容的不同。
- -v或--version　显示版本信息。
- -w或--ignore-all-space　忽略全部的空格字符。
- -W<宽度>或--width<宽度>　在使用-y参数时，指定栏宽。
- -x<文件名或目录>或--exclude<文件名或目录>　不比较选项中所指定的文件或目录。
- -X<文件>或--exclude-from<文件>　您可以将文件或目录类型存成文本文件，然后在=<文件>中指定此文本文件。
- -y或--side-by-side　以并列的方式显示文件的异同之处。
- --help　显示帮助。
- --left-column　在使用-y参数时，若两个文件某一行内容相同，则仅在左侧的栏位显示该行内容。
- --suppress-common-lines　在使用-y参数时，仅显示不同之处。

实例1：比较两个文件

```
[root@localhost test3]# diff log2014.log log2013.log
3c3
< 2014-03
---
> 2013-03
8c8
< 2013-07
---
> 2013-08
11,12d10
< 2013-11
< 2013-12
```

上面的"3c3"和"8c8"表示log2014.log和log20143log文件在3行和第8行内容有所不同；"11,12d10"表示第一个文件比第二个文件多了第11和12行。

实例2：并排格式输出

```
[root@localhost test3]# diff log2014.log log2013.log  -y -W 50
2013-01                 2013-01
2013-02                 2013-02
2014-03               | 2013-03
2013-04                 2013-04
2013-05                 2013-05
2013-06                 2013-06
2013-07                 2013-07
2013-07               | 2013-08
2013-09                 2013-09
2013-10                 2013-10
2013-11               <
2013-12               <
[root@localhost test3]# diff log2013.log log2014.log  -y -W 50
2013-01                 2013-01
2013-02                 2013-02
2013-03               | 2014-03
2013-04                 2013-04
2013-05                 2013-05
2013-06                 2013-06
2013-07                 2013-07
2013-08               | 2013-07
2013-09                 2013-09
2013-10                 2013-10
                      > 2013-11
                      > 2013-12
```

说明：

- "|"表示前后2个文件内容有不同
- "<"表示后面文件比前面文件少了1行内容
- ">"表示后面文件比前面文件多了1行内容

Linux diffstat命令

Linux diffstat命令根据diff的比较结果，显示统计数字。

diffstat读取diff的输出结果，然后统计各文件的插入，删除，修改等差异计量。

语法

```
diff [-wV][-n <文件名长度>][-p <文件名长度>]
```

参数：

- -n<文件名长度>　指定文件名长度，指定的长度必须大于或等于所有文件中最长的文件名。
- -p<文件名长度>　与-n参数相同，但此处的<文件名长度>包括了文件的路径。
- -w　指定输出时栏位的宽度。
- -V　显示版本信息。

实例

用户也可以直接使用"|"将diff指令所输出的结果直接送给diffstat指令进行统计结果的显示。

使用该指令时，若所比较的文件或者子目录不在当前目录下，则应该使用其完整路径。

将目录"test1"和"test2"下的同名文件"testf.txt"使用diff指令进行比较。然后使用diffstat指令对结果进行统计显示，输入如下命令：

```
$ diff test1 test2 | diffstat   #进行比较结果的统计显示
```

注意：使用这条命令可以非常方便地实现统计显示的功能。

对于查看文件中的内容，用户可以通过指令"cat"进行查看即可，具体操作如下：

```
$ cat test1/testf.txt           #查看test1/testf的内容
abc
def
ghi
jkl
mno
pqr
stu
vws
$ cat test2/testf.txt           #查看test2/testf的内容
abc
def
ghi
jkl
mno
```

从上面的文件内容显示，可以看到两个文件内容的差别。现在来运行刚才的命令，对文件比较的结果进行统计显示，结果如下：

```
testfile | 2 +-             #统计信息输出显示
1 file changed, 1 insertion(+), 1 deletion(-)
```

Linux file命令

Linux file命令用于辨识文件类型。

通过file指令，我们得以辨识该文件的类型。

语法

```
file [-beLvz][-f <名称文件>][-m <魔法数字文件>...][文件或目录...]
```

参数：

- -b 列出辨识结果时，不显示文件名称。
- -c 详细显示指令执行过程，便于排错或分析程序执行的情形。
- -f<名称文件> 指定名称文件，其内容有一个或多个文件名称呢感，让file依序辨识这些文件，格式为每列一个文件名称。
- -L 直接显示符号连接所指向的文件的类别。
- -m<魔法数字文件> 指定魔法数字文件。
- -v 显示版本信息。
- -z 尝试去解读压缩文件的内容。
- [文件或目录...] 要确定类型的文件列表，多个文件之间使用空格分开，可以使用shell通配符匹配多个文件。

实例

显示文件类型：

```
[root@localhost ~]# file install.log
install.log: UTF-8 Unicode text

[root@localhost ~]# file -b install.log      <== 不显示文件名称
UTF-8 Unicode text

[root@localhost ~]# file -i install.log      <== 显示MIME类别。
install.log: text/plain; charset=utf-8

[root@localhost ~]# file -b -i install.log
text/plain; charset=utf-8
```

显示符号链接的文件类型

```
[root@localhost ~]# ls -l /var/mail
lrwxrwxrwx 1 root root 10 08-13 00:11 /var/mail -> spool/mail

[root@localhost ~]# file /var/mail
/var/mail: symbolic link to `spool/mail'

[root@localhost ~]# file -L /var/mail
/var/mail: directory

[root@localhost ~]# file /var/spool/mail
/var/spool/mail: directory

[root@localhost ~]# file -L /var/spool/mail
/var/spool/mail: directory
```

Linux find命令

Linux find命令用来在指定目录下查找文件。任何位于参数之前的字符串都将被视为欲查找的目录名。如果使用该命令时，不设置任何参数，则find命令将在当前目录下查找子目录与文件。并且将查找到的子目录和文件全部进行显示。

语法

```
find   path   -option   [   -print ]   [ -exec   -ok   command ]   {} ;
```

参数说明 :

find 根据下列规则判断 path 和 expression，在命令列上第一个 - ( ) , ! 之前的部份为 path，之后的是 expression。如果 path 是空字串则使用目前路径，如果 expression 是空字串则使用 -print 为预设 expression。

expression 中可使用的选项有二三十个之多，在此只介绍最常用的部份。

-mount, -xdev : 只检查和指定目录在同一个文件系统下的文件，避免列出其它文件系统中的文件

-amin n : 在过去 n 分钟内被读取过

-anewer file : 比文件 file 更晚被读取过的文件

-atime n : 在过去 n 天过读取过的文件

-cmin n : 在过去 n 分钟内被修改过

-cnewer file :比文件 file 更新的文件

-ctime n : 在过去 n 天过修改过的文件

-empty : 空的文件-gid n or -group name : gid 是 n 或是 group 名称是 name

-ipath p, -path p : 路径名称符合 p 的文件，ipath 会忽略大小写

-name name, -iname name : 文件名称符合 name 的文件。iname 会忽略大小写

-size n : 文件大小 是 n 单位，b 代表 512 位元组的区块，c 表示字元数，k 表示 kilo bytes，w 是二个位元组。-type c : 文件类型是 c 的文件。

d: 目录

c: 字型装置文件

b: 区块装置文件

p: 具名贮列

f: 一般文件

l: 符号连结

s: socket

-pid n : process id 是 n 的文件

你可以使用 ( ) 将运算式分隔，并使用下列运算。

exp1 -and exp2

! expr

-not expr

exp1 -or exp2

exp1, exp2

实例

将目前目录及其子目录下所有延伸档名是 c 的文件列出来。

```
# find . -name "*.c"
```

将目前目录其其下子目录中所有一般文件列出

```
# find . -ftype f
```

将目前目录及其子目录下所有最近 20 分钟内更新过的文件列出

```
# find . -ctime -20
```

查找/var/logs目录中更改时间在7日以前的普通文件，并在删除之前询问它们：

```
$ find /var/logs -type f -mtime +7 -ok rm { } ;
```

查找前目录中文件属主具有读、写权限，并且文件所属组的用户和其他用户具有读权限的文件：

```
$ find . -type f -perm 644 -exec ls -l { } ;
```

为了查找系统中所有文件长度为0的普通文件，并列出它们的完整路径：

```
$ find / -type f -size 0 -exec ls -l { } ;
```

查找/var/logs目录中更改时间在7日以前的普通文件，并在删除之前询问它们:

```
$ find /var/logs -type f -mtime +7 -ok rm { } ;
```

Linux git命令

Linux git命令是文字模式下的文件管理员。

git是用来管理文件的程序，它十分类似DOS下的Norton Commander，具有互动式操作界面。它的操作方法和Norton Commander几乎一样。

语法

```
git
```

操作说明：

- F1 ：执行info指令，查询指令相关信息，会要求您输入欲查询的名称。
- F2 ：执行cat指令，列出文件内容。
- F3 ：执行gitview指令，观看文件内容。
- F4 ：执行vi指令，编辑文件内容。
- F5 ：执行cp指令，复制文件或目录，会要求您输入目标文件或目录。
- F6 ：执行mv指令，移动文件或目录，或是更改其名称，会要求您输入目标文件或目录。
- F7 ：执行mkdir指令，建立目录。
- F8 ：执行rm指令，删除文件或目录。
- F9 ：执行make指令，批处理执行指令或编译程序时，会要求您输入相关命令。
- F10 ：离开git文件管理员。

Linux gitview命令

Linux gitview命令用于观看文件的内容，它会同时显示十六进制和ASCII格式的字码。

语法

```
gitview [-bchilv][文件]
```

参数：

- -b　单色模式，不使用ANSI控制码显示彩色。
- -c　彩色模式，使用ANSI控制码显示色彩。
- -h　在线帮助。
- -i　显示存放gitview程序的所在位置。
- -l　不使用先前的显示字符。
- -v　显示版本信息。

实例

使用指令gitview以彩色模式观看文件"/home/ rootlocal/demo.txt"中的内容，输入如下命令：

```
$ gitview -c /home/rootlocal/demo.txt      #使用gitview指令观看指定文件内容
```

Linux indent命令

Linux indent命令用于调整C原始代码文件的格式。

indent可辨识C的原始代码文件，并加以格式化，以方便程序设计师阅读。

语法

```
indent [参数][源文件] 或 indent [参数][源文件][-o 目标文件]
```

参数：

- -bad或--blank-lines-after-declarations 　在声明区段或加上空白行。
- -bap或--blank-lines-after-procedures 　在程序或加上空白行。
- -bbb或--blank-lines-after-block-comments 　在注释区段后加上空白行。
- -bc或--blank-lines-after-commas 　在声明区段中，若出现逗号即换行。
- -bl或--braces-after-if-line 　if(或是else,for等等)与后面执行区段的"{"不同行，且"}"自成一行。
- -bli<缩排格数>或--brace-indent<缩排格数> 　设置{ }缩排的格数。
- -br或--braces-on-if-line 　if(或是else,for等等)与后面执行跛段的"{"不同行，且"}"自成一行。
- -bs或--blank-before-sizeof 　在sizeof之后空一格。
- -c<栏数>或--comment-indentation<栏数> 　将注释置于程序码右侧指定的栏位。
- -cd<栏数>或--declaration-comment-column<栏数> 　将注释置于声明右侧指定的栏位。
- -cdb或--comment-delimiters-on-blank-lines 　注释符号自成一行。
- -ce或--cuddle-else 　将else置于"}"(if执行区段的结尾)之后。
- -ci<缩排格数>或--continuation-indentation<缩排格数> 　叙述过长而换行时，指定换行后缩排的格数。
- -cli<缩排格数>或--case-indentation-<缩排格数> 　使用case时，switch缩排的格数。
- -cp<栏数>或-else-endif-column<栏数> 　将注释置于else与elseif叙述右侧定的栏位。
- -cs或--space-after-cast 　在cast之后空一格。
- -d<缩排格数>或-line-comments-indentation<缩排格数> 　针对不是放在程序码右侧的注释，设置其缩排格数。
- -di<栏数>或--declaration-indentation<栏数> 　将声明区段的变量置于指定的栏位。
- -fc1或--format-first-column-comments 　针对放在每行最前端的注释，设置其格式。
- -fca或--format-all-comments 　设置所有注释的格式。
- -gnu或--gnu-style 　指定使用GNU的格式，此为预设值。
- -i<格数>或--indent-level<格数> 　设置缩排的格数。
- -ip<格数>或--parameter-indentation<格数> 　设置参数的缩排格数。

- -kr或--k-and-r-style 指定使用Kernighan&Ritchie的格式。
- -lp或--continue-at-parentheses 叙述过长而换行，且叙述中包含了括弧时，将括弧中的每行起始栏位内容垂直对其排列。
- -nbad或--no-blank-lines-after-declarations 在声明区段后不要加上空白行。
- -nbap或--no-blank-lines-after-procedures 在程序后不要加上空白行。
- -nbbb或--no-blank-lines-after-block-comments 在注释区段后不要加上空白行。
- -nbc或--no-blank-lines-after-commas 在声明区段中，即使出现逗号，仍旧不要换行。
- -ncdb或--no-comment-delimiters-on-blank-lines 注释符号不要自成一行。
- -nce或--dont-cuddle-else 不要将else置于"}"之后。
- -ncs或--no-space-after-casts 不要在cast之后空一格。
- -nfc1或--dont-format-first-column-comments 不要格式化放在每行最前端的注释。
- -nfca或--dont-format-comments 不要格式化任何的注释。
- -nip或--no-parameter-indentation 参数不要缩排。
- -nlp或--dont-line-up-parentheses 叙述过长而换行，且叙述中包含了括弧时，不用将括弧中的每行起始栏位垂直对其排列。
- -npcs或--no-space-after-function-call-names 在调用的函数名称之后，不要加上空格。
- -npro或--ignore-profile 不要读取indent的配置文件.indent.pro。
- -npsl或--dont-break-procedure-type 程序类型与程序名称放在同一行。
- -nsc或--dont-star-comments 注解左侧不要加上星号(*)。
- -nsob或--leave-optional-semicolon 不用处理多余的空白行。
- -nss或--dont-space-special-semicolon 若for或while区段仅有一行时，在分号前不加上空格。
- -nv或--no-verbosity 不显示详细的信息。
- -orig或--original 使用Berkeley的格式。
- -pcs或--space-after-procedure-calls 在调用的函数名称与"{"之间加上空格。
- -psl或--procnames-start-lines 程序类型置于程序名称的前一行。
- -sc或--start-left-side-of-comments 在每行注释左侧加上星号(*)。
- -sob或--swallow-optional-blank-lines 删除多余的空白行。
- -ss或--space-special-semicolon 若for或swile区段今有一行时，在分号前加上空格。
- -st或--standard-output 将结果显示在标准输出设备。
- -T 数据类型名称缩排。
- -ts<格数>或--tab-size<格数> 设置tab的长度。
- -v或--verbose 执行时显示详细的信息。
- -version 显示版本信息。

Indent代码格式化说明

使用的**indent**参数	值	含义
--blank-lines-after-declarations	bad	变量声明后加空行

--blank-lines-after-procedures	bap	函数结束后加空行
--blank-lines-before-block-comments	bbb	块注释前加空行
--break-before-boolean-operator	bbo	较长的行，在逻辑运算符前分行
--blank-lines-after-commas	nbc	变量声明中，逗号分隔的变量不分行
--braces-after-if-line	bl	"if"和"{"分做两行
--brace-indent 0	bli0	"{"不继续缩进
--braces-after-struct-decl-line	bls	定义结构，"struct"和"{"分行
--comment-indentationn	c33	语句后注释开始于行33
--declaration-comment-columnn	cd33	变量声明后注释开始于行33
--comment-delimiters-on-blank-lines	ncdb	不将单行注释变为块注释
--cuddle-do-while	ncdw	"do --- while"的"while"和其前面的"}"另起一行
--cuddle-else	nce	"else"和其前面的"}"另起一行
--case-indentation 0	cli0	switch中的case语句所进0个空格
--else-endif-columnn	cp33	#else, #endif后面的注释开始于行33
--space-after-cast	cs	在类型转换后面加空格
--line-comments-indentation n	d0	单行注释（不从1列开始的），不向左缩进
--break-function-decl-args	nbfda	关闭：函数的参数一个一行
--declaration-indentationn	di2	变量声明，变量开始于2行，即不必对齐
--format-first-column-comments	nfc1	不格式化起于第一行的注释
--format-all-comments	nfca	不开启全部格式化注释的开关
--honour-newlines	hnl	Prefer to break long lines at the position of newlines in the input.
--indent-leveln	i4	设置缩进多少字符，如果为tab的整数倍，用tab来缩进，否则用空格填充。
--parameter-indentationn	ip5	旧风格的函数定义中参数说明缩进5个空格
--line-length 75	l75	非注释行最长75

--continue-at-parentheses	lp	续行从上一行出现的括号开始
--space-after-procedure-calls	pcs	函数和"("之间插入一个空格
--space-after-parentheses	nprs	在"（"后"）"前不插入空格
--procnames-start-lines	psl	将函数名和返回类型放在两行定义
--space-after-for	saf	for后面有空格
--space-after-if	sai	if后面有空格
--space-after-while	saw	while后面有空格
--start-left-side-of-comments	nsc	不在生成的块注释中加*
--swallow-optional-blank-lines	nsob	不去掉可添加的空行
--space-special-semicolon	nss	一行的for或while语句，在";"前不加空。
--tab-size	ts4	一个tab为4个空格（要能整除"-in"）
--use-tabs	ut	使用tab来缩进

Linux cut命令

Linux cut命令用于显示每行从开头算起 num1 到 num2 的文字。

语法

```
cut  [-bn] [file]
cut [-c] [file]
cut [-df] [file]
```

使用说明:

cut 命令从文件的每一行剪切字节、字符和字段并将这些字节、字符和字段写至标准输出。

如果不指定 File 参数，cut 命令将读取标准输入。必须指定 -b、-c 或 -f 标志之一。

参数:

- -b ：以字节为单位进行分割。这些字节位置将忽略多字节字符边界，除非也指定了 -n 标志。
- -c ：以字符为单位进行分割。
- -d ：自定义分隔符，默认为制表符。
- -f ：与-d一起使用，指定显示哪个区域。
- -n ：取消分割多字节字符。仅和 -b 标志一起使用。如果字符的最后一个字节落在由 -b 标志的 List 参数指示的
  范围之内，该字符将被写出；否则，该字符将被排除

实例

当你执行who命令时，会输出类似如下的内容：

```
$ who
rocrocket :0           2009-01-08 11:07
rocrocket pts/0        2009-01-08 11:23 (:0.0)
rocrocket pts/1        2009-01-08 14:15 (:0.0)
```

如果我们想提取每一行的第3个字节，就这样：

```
$ who|cut -b 3
c
c
```

Linux ln命令

Linux ln命令是一个非常重要命令，它的功能是为某一个文件在另外一个位置建立一个同步的链接。

当我们需要在不同的目录，用到相同的文件时，我们不需要在每一个需要的目录下都放一个必须相同的文件，我们只要在某个固定的目录，放上该文件，然后在 其它的目录下用ln命令链接（link）它就可以，不必重复的占用磁盘空间。

语法

```
ln [参数][源文件或目录][目标文件或目录]
```

其中参数的格式为

[-bdfinsvF] [-S backup-suffix] [-V {numbered,existing,simple}]

[--help] [--version] [--]

命令功能 :
Linux文件系统中，有所谓的链接(link)，我们可以将其视为档案的别名，而链接又可分为两种 : 硬链接(hard link)与软链接(symbolic link)，硬链接的意思是一个档案可以有多个名称，而软链接的方式则是产生一个特殊的档案，该档案的内容是指向另一个档案的位置。硬链接是存在同一个文件系统中，而软链接却可以跨越不同的文件系统。

不论是硬链接或软链接都不会将原本的档案复制一份，只会占用非常少量的磁碟空间。

软链接：

- 1.软链接，以路径的形式存在。类似于Windows操作系统中的快捷方式
- 2.软链接可以 跨文件系统 ，硬链接不可以
- 3.软链接可以对一个不存在的文件名进行链接
- 4.软链接可以对目录进行链接

硬链接：

- 1.硬链接，以文件副本的形式存在。但不占用实际空间。
- 2.不允许给目录创建硬链接
- 3.硬链接只有在同一个文件系统中才能创建

命令参数

必要参数：

- -b 删除，覆盖以前建立的链接
- -d 允许超级用户制作目录的硬链接
- -f 强制执行
- -i 交互模式，文件存在则提示用户是否覆盖
- -n 把符号链接视为一般目录
- -s 软链接(符号链接)
- -v 显示详细的处理过程

选择参数：

- -S "-S<字尾备份字符串> "或 "--suffix=<字尾备份字符串>"
- -V "-V<备份方式>"或"--version-control=<备份方式>"
- --help 显示帮助信息
- --version 显示版本信息

实例

给文件创建软链接，为log2013.log文件创建软链接link2013，如果log2013.log丢失，link2013将失效：

```
ln -s log2013.log link2013
```

输出：

```
[root@localhost test]# ll
-rw-r--r-- 1 root bin      61 11-13 06:03 log2013.log
[root@localhost test]# ln -s log2013.log link2013
[root@localhost test]# ll
lrwxrwxrwx 1 root root     11 12-07 16:01 link2013 -> log2013.log
-rw-r--r-- 1 root bin      61 11-13 06:03 log2013.log
```

给文件创建硬链接，为log2013.log创建硬链接ln2013，log2013.log与ln2013的各项属性相同

```
ln log2013.log ln2013
```

输出：

```
[root@localhost test]# ll
lrwxrwxrwx 1 root root     11 12-07 16:01 link2013 -> log2013.log
-rw-r--r-- 1 root bin      61 11-13 06:03 log2013.log
[root@localhost test]# ln log2013.log ln2013
[root@localhost test]# ll
lrwxrwxrwx 1 root root     11 12-07 16:01 link2013 -> log2013.log
-rw-r--r-- 2 root bin      61 11-13 06:03 ln2013
-rw-r--r-- 2 root bin      61 11-13 06:03 log2013.log
```

Linux less命令

less 与 more 类似，但使用 less 可以随意浏览文件，而 more 仅能向前移动，却不能向后移动，而且 less 在查看之前不会加载整个文件。

语法

```
less [参数] 文件
```

参数说明：

- -b <缓冲区大小> 设置缓冲区的大小
- -e 当文件显示结束后，自动离开
- -f 强迫打开特殊文件，例如外围设备代号、目录和二进制文件
- -g 只标志最后搜索的关键词
- -i 忽略搜索时的大小写
- -m 显示类似more命令的百分比
- -N 显示每行的行号
- -o <文件名> 将less 输出的内容在指定文件中保存起来
- -Q 不使用警告音
- -s 显示连续空行为一行
- -S 行过长时间将超出部分舍弃
- -x <数字> 将"tab"键显示为规定的数字空格
- /字符串：向下搜索"字符串"的功能
- ?字符串：向上搜索"字符串"的功能
- n：重复前一个搜索（与 / 或 ? 有关）
- N：反向重复前一个搜索（与 / 或 ? 有关）
- b 向后翻一页
- d 向后翻半页
- h 显示帮助界面
- Q 退出less 命令
- u 向前滚动半页
- y 向前滚动一行
- 空格键 滚动一行
- 回车键 滚动一页
- [pagedown]： 向下翻动一页
- [pageup]： 向上翻动一页

实例

1、查看文件

```
less log2013.log
```

2、ps查看进程信息并通过less分页显示

```
ps -ef |less
```

3、查看命令历史使用记录并通过less分页显示

```
[root@localhost test]# history | less
22  scp -r tomcat6.0.32 root@192.168.120.203:/opt/soft
23  cd ..
24  scp -r web root@192.168.120.203:/opt/
25  cd soft
26  ls
……省略……
```

4、浏览多个文件

```
less log2013.log log2014.log
```

说明：
输入 ：n后，切换到 log2014.log
输入 ：p 后，切换到log2013.log

附加备注

1.全屏导航

- ctrl + F - 向前移动一屏
- ctrl + B - 向后移动一屏
- ctrl + D - 向前移动半屏
- ctrl + U - 向后移动半屏

2.单行导航

- j - 向前移动一行
- k - 向后移动一行

3.其它导航

- G - 移动到最后一行
- g - 移动到第一行

- q / ZZ - 退出 less 命令

4.其它有用的命令

- v - 使用配置的编辑器编辑当前文件
- h - 显示 less 的帮助文档
- &pattern - 仅显示匹配模式的行，而不是整个文件

5.标记导航

当使用 less 查看大文件时，可以在任何一个位置作标记，可以通过命令导航到标有特定标记的文本位置：

- ma - 使用 a 标记文本的当前位置
- 'a - 导航到标记 a 处

Linux locate命令

Linux locate命令用于查找符合条件的文档，他会去保存文档和目录名称的数据库内，查找合乎范本样式条件的文档或目录。

一般情况我们只需要输入 **locate your_file_name** 即可查找指定文件。

语法

```
locate [-d ][--help][--version][范本样式...]
```

参数：

- -d或--database= 配置locate指令使用的数据库。locate指令预设的数据库位于/var/lib/slocate目录里，文档名为slocate.db，您可使用 这个参数另行指定。
- --help　在线帮助。
- --version　显示版本信息。

实例

查找passwd文件，输入以下命令：

```
locate passwd
```

附加说明

locate与find 不同: find 是去硬盘找，locate 只在/var/lib/slocate资料库中找。

locate的速度比find快，它并不是真的查找，而是查数据库，一般文件数据库在/var/lib/slocate/slocate.db中，所以locate的查找并不是实时的，而是以数据库的更新为准，一般是系统自己维护，也可以手工升级数据库，命令为：

```
locate -u
```

Linux lsattr命令

Linux lsattr命令用于显示文件属性。

用chattr执行改变文件或目录的属性，可执行lsattr指令查询其属性。

语法

```
lsattr [-adlRvV][文件或目录...]
```

参数：

- -a 显示所有文件和目录，包括以"."为名称开头字符的额外内建，现行目录"."与上层目录".."。
- -d 显示，目录名称，而非其内容。
- -l 此参数目前没有任何作用。
- -R 递归处理，将指定目录下的所有文件及子目录一并处理。
- -v 显示文件或目录版本。
- -V 显示版本信息。

实例

1、用chattr命令防止系统中某个关键文件被修改：

```
# chattr +i /etc/resolv.conf
```

然后用mv /etc/resolv.conf等命令操作于该文件，都是得到Operation not permitted 的结果。

vim编辑该文件时会提示W10: Warning: Changing a readonly file错误。要想修改此文件就要把i属性去掉：

```
chattr -i /etc/resolv.conf
```

使用 lsattr 命令来显示文件属性：

```
# lsattr /etc/resolv.conf
```

输出结果为:

```
----i-------- /etc/resolv.conf
```

2、让某个文件只能往里面追加数据，但不能删除，适用于各种日志文件：

```
# chattr +a /var/log/messages
```

Linux mattrib命令

Linux mattrib命令用来变更或显示MS-DOS文件的属性。

mattrib为mtools工具指令，模拟MS-DOS的attrib指令，可变更MS-DOS文件的属性。

语法

```
mattrib [-a|+a] [-h|+h] [-r|+r] [-s|+s] [-/] [-X] msdosfile [ msdosfiles ... ]
```

参数：

- -a/+a 除去/设定备份属性。
- -h/+h 除去/设定隐藏属性。
- -r/+r 除去/设定唯读属性。
- -s/+s 除去/设定系统属性。
- -/ 递回的处理包含所有子目录下的档案。
- -X 以较短的格式输出结果。

实例

列出 A 槽 MSDOS 格式磁片上所有文件的属性。

```
mattrib a:
```

除去 A 槽磁片上 msdos.sys 档案的隐藏、系统与唯读属性。

```
mattrib -h -s -r a:msdos.sys
```

除去 A 槽磁片上包含子目录下所有档案的唯读属性。

```
mattrib -r -/ a:*.*
```

Linux mc命令

Linux mc命令用于提供一个菜单式的文件管理程序。

执行mc之后，将会看到菜单式的文件管理程序，共分成 4 个部分。

语法

```
mc [-abcdfhkPstuUVx][-C <参数>][-l <文件>][-v <文件>][目录]
```

参 数：

- -a 当mc程序画线时不用绘图字符画线。
- -b 使用单色模式显示。
- -c 使用彩色模式显示。
- -C<参数> 指定显示的颜色。
- -d 不使用鼠标。
- -f 显示mc函数库所在的目录。
- -h 显示帮助。
- -k 重设softkeys成预设置。
- -l<文件> 在指定文件中保存ftpfs对话窗的内容。
- -P 程序结束时，列出最后的工作目录。
- -s 用慢速的终端机模式显示，在这模式下将减少大量的绘图及文字显示。
- -t 使用TEMPCAP变量设置终端机，而不使用预设置。
- -u 不用目前的shell程序。
- -U 使用目前的shell程序。
- -v<文件> 使用mc的内部编辑器来显示指定的文件。
- -V 显示版本信息。
- -x 指定以xterm模式显示。

Linux MC 相关操作

命令按键	描 述
F9 or Esc+9	激活菜单栏
Tab	在两个窗口间移动
F10 or Esc+0	退出MC
Control-Enter or Alt-Enter	可以将文件名拷贝到命令行
F1 or Esc+1	打开帮助页面

虽然MC很好用，不过我还是建议大家使用命令行工具！

Linux mdel命令

Linux mdel命令用来删除 MSDOS 格式的档案。

在删除只读之前会有提示信息产生。

语法

```
mdel [-v] msdosfile [ msdosfiles ... ]
```

参数：

- -v 显示更多的讯息。

实例

将 A 槽磁片根目录中的 autoexec.bat 删除。

```
mdel a:autoexec.bat .
```

Linux mdir命令

Linux mdir命令用于显示MS-DOS目录。

mdir为mtools工具指令，模拟MS-DOS的dir指令，可显示MS-DOS文件系统中的目录内容。

语法

```
mdir [-afwx/][目录]
```

参数：

- -/ 显示目录下所有子目录与文件。
- -a 　显示隐藏文件。
- -f 　不显示磁盘所剩余的可用空间。
- -w 　仅显示目录或文件名称，并以横排方式呈现，以便一次能显示较多的目录或文件。
- -X 　仅显示目录下所有子目录与文件的完整路径，不显示其他信息。

实例

显示a盘中的内容

```
$ mdir -/ a:\*
```

以上命令执行后，mdir将显示指定盘"a:\"中的所有子目录及其中的文件信息，如下所示：

```
Volume in drive A has no label  #加载信息
Volume Serial Number is 13D2~055C
Directory for A:\                   #以下为目录信息
./TEST <DIR> 2011-08-23 16:59
#显示格式为文件名，目录大小，修改时间
AUTORUN.INF 265 2011-08-23 16:53
AUTORUN.BAT 43 2011-08-23 16:56
3 files 308 bytes               #统计总大小
724 325 bytes free              #剩余空间
```

Linux mktemp命令

Linux mktemp命令用于建立暂存文件。

mktemp建立的一个暂存文件，供shell script使用。

语法

```
mktemp [-qu][文件名参数]
```

参数：

- -q 　执行时若发生错误，不会显示任何信息。
- -u 　暂存文件会在mktemp结束前先行删除。
- [文件名参数] 　文件名参数必须是以"自订名称.XXXXXX"的格式。

实例

使用mktemp 命令生成临时文件时，文件名参数应当以"文件名.XXXX"的形式给出，mktemp 会根据文件名参数建立一个临时文件。在命令行提示符输入如下命令：

```
mktemp tmp.xxxx #生成临时文件
```

使用该命令后，可使用dir 或ls看当前目录，得到如下结果：

```
cmd@cmd-desktop:~$ mktemp tmp.xxxx #生成临时文件
cmd@cmd-desktop:~$dir #查看当前目录
file test testfile testfile1 tmp.3847 #生成了tmp.3847
```

由此可见，生成的临时文件为tmp.3847，其中，文件名参数中的"XXXX"被4 个随机产生的字符所取代。

Linux more命令

Linux more 命令类似 cat ，不过会以一页一页的形式显示，更方便使用者逐页阅读，而最基本的指令就是按空白键（space）就往下一页显示，按 b 键就会往回（back）一页显示，而且还有搜寻字串的功能（与 vi 相似），使用中的说明文件，请按 h 。

语法

```
more [-dlfpcsu] [-num] [+/pattern] [+linenum] [fileNames..]
```

参数：

- -num 一次显示的行数
- -d 提示使用者，在画面下方显示 [Press space to continue, 'q' to quit.] ，如果使用者按错键，则会显示 [Press 'h' for instructions.] 而不是 '哔' 声
- -l 取消遇见特殊字元 ^L（送纸字元）时会暂停的功能
- -f 计算行数时，以实际上的行数，而非自动换行过后的行数（有些单行字数太长的会被扩展为两行或两行以上）
- -p 不以卷动的方式显示每一页，而是先清除萤幕后再显示内容
- -c 跟 -p 相似，不同的是先显示内容再清除其他旧资料
- -s 当遇到有连续两行以上的空白行，就代换为一行的空白行
- -u 不显示下引号 （根据环境变数 TERM 指定的 terminal 而有所不同）
- +/pattern 在每个文档显示前搜寻该字串（pattern），然后从该字串之后开始显示
- +num 从第 num 行开始显示
- fileNames 欲显示内容的文档，可为复数个数

实例

逐页显示 testfile 文档内容，如有连续两行以上空白行则以一行空白行显示。

```
more -s testfile
```

从第 20 行开始显示 testfile 之文档内容。

```
more +20 testfile
```

常用操作命令

- Enter 向下n行，需要定义。默认为1行

- Ctrl+F 向下滚动一屏
- 空格键 向下滚动一屏
- Ctrl+B 返回上一屏
- = 输出当前行的行号
- ：f 输出文件名和当前行的行号
- V 调用vi编辑器
- !命令 调用Shell，并执行命令
- q 退出more

Linux mmove命令

Linux mmove命令用于在MS-DOS文件系统中，移动文件或目录，或更改名称。

mmove为mtools工具命令，模拟MS-DOS的move命令，可在MS-DOS文件系统中移动现有的文件或目录，或是更改现有文件或目录的名称。

语法

```
mmove [源文件或目录...][目标文件或目录]
```

参数说明:

- [源文件或目录…]: 执行操作的源文件或目录路径
- [目标文件或目录]: 执行操作后的目标文件或目录路径

实例

使用指令mmove将文件"autorun.bat"移动到目录"test"中，输入如下命令：

```
$ mmove autorun.bat test        #移动文件到目录test中
```

以上命令执行以后，指令mmove会将文件"autorun.bat"移动到指定目录"test"中。

注意：用户可以使用mdir指令查看移动后的文件或目录信息。

Linux mread命令

Linux mread命令用于将MS-DOS文件复制到Linux/Unix的目录中。

mread为mtools工具命令，可将MS-DOS文件复制到Linux的文件系统中。这个命令目前已经不常用，一般都使用mcopy命令来代替。

语法

```
mread [MS-DOS文件...][Linux文件或目录]
```

参数说明:

- [MS-DOS文件...]: 执行操作的DOS源文件或目录路径
- [Linux文件或目录]: 执行操作后的Linux目标文件或目录路径

实例

使用指令mread将盘"a:\"中的所有内容复制到当前工作目录下，输入如下命令：

```
$ mread a:\* ./     #将a盘上的所有文件复制到当前工作目录
```

执行该命令前，可以先使用mdir命令查看原来的目录结构。执行mread之后，可使用ls命令再次查看复制之后的文件结构，结果如下所示：

```
$ mdir -/ a:\*      #查看a盘中的文件
Volume in drive A has no label          #加载信息
Volume Serial Number is 13D2~055C
Directory for A:/                   #以下为目录信息
./TEST <DIR> 2011-08-23 16:59
#显示格式为文件名，目录大小，修改时间
AUTORUN.INF 265 2011-08-23 16:53
AUTORUN.BAT 43 2011-08-23 16:56
3 files 308 bytes                       #统计总大小
724 325 bytes free                  #剩余空间
$ mread A:\* ./     #将a盘上所有文件复制到当前工作目录
$ ls                                    #查看文件或子目录信息
TEST AUTORUN.INF AUTORUN.BAT            #显示复制后的内容
```

Linux mren命令

Linux mren命令用于更改MS-DOS文件或目录的名称，或是移动文件或目录。

mren为MS-DOS工具指令，与DOS下的ren指令相似，可以实现更改MS-DOS文件或目录名称。

源文件必须是磁盘上已经存在的文件，若忽略盘符及路径，则表示当前盘及当前目录的文件。

新文件名是所要更换的文件名称。新文件名称前不可以加与源文件不同的盘符及路径，因为该命令只能更改同一盘上的文件名称。

语法

```
mren [源文件或目录...][目标文件或目录]
```

参数说明：

- [源文件或目录...]： 执行操作的源文件名或者源文件路径

实例

使用指令mren将a盘下的文件"autorun.bat"的文件名修改为"auto.bat"，输入如下命令：

```
$ mren a:\autorun.bat auto.bat
#将文件autorun.bat重命名为auto.bat
```

使用该命令前后使用mdir命令查看并对比，得到结果如下：

```
$ mdir -/ a:\*                    #查看a盘中的文件
Volume in drive A has no label  #加载信息
Volume Serial Number is 13D2~055C
Directory for A:\                    #以下为目录信息
./TEST <DIR> 2011-08-23 16:59       #文件名，目录大小，修改时间
AUTORUN.BAT 43 2011-08-23 16:56
3 files 308 bytes                    #统计总大小
724 325 bytes free               #剩余空间
#将文件autorun.bat重命名为auto.bat
$ mren a:\autorun.bat auto.bat
$ mdir -/ a:\*                    #再次查看a盘中文件
Volume in drive A has no label      #加载信息
Volume Serial Number is 13D2~055C
Directory for A:\                    #以下为目录信息
./TEST <DIR> 2011-08-23 16:59       #文件名目录大小 修改时间
#文件名被改为auto.bat，修改时间改为当前系统时间
AUTO.BAT 43 2011-08-23 16:56
3 files 308 bytes                    #统计总大小
724 325 bytes free               #剩余空间
```

Linux mtools命令

Linux mtools命令用于显示mtools支持的指令。

mtools为MS-DOS文件系统的工具程序，可模拟许多MS-DOS的指令。这些指令都是mtools的符号连接，因此会有一些共同的特性。

语法

```
mtools
```

参数说明：

- -a 长文件名重复时自动更改目标文件的长文件名。
- -A 短文件名重复但长文件名不同时自动更改目标文件的短文件名。
- -o 长文件名重复时，将目标文件覆盖现有的文件。
- -O 短文件名重复但长文件名不同时，将目标文件覆盖现有的文件。
- -r 长文件名重复时，要求用户更改目标文件的长文件名。
- -R 短文件名重复但长文件名不同时，要求用户更改目标文件的短文件名。
- -s 长文件名重复时，则不处理该目标文件。
- -S 短文件名重复但长文件名不同时，则不处理该目标文件。
- -v 执行时显示详细的说明。
- -V 显示版本信息。

实例

显示 mtools软件包所支持的MS-DOS命令。

在命令提示符中直接输入mtools，可显示其所支持的MS-DOS命令，如下所示：

```
$ mtools #显示所支持的MS-DOS命令
Supported commands: #命令列表
mattrib, mbadblocks, mcat, mcd, mclasserase, mcopy, mdel, mdeltree
mdir, mdoctorfat, mdu, mformat, minfo, mlabel, mmd, mmount
mpartition, mrd, mread, mmove, mren, mshowfat, mtoolstest, mtype
mwrite, mzip
```

Linux mtoolstest命令

Linux mtoolstest命令用于测试并显示mtools的相关设置。

mtoolstest为mtools工具指令，可读取与分析mtools的配置文件，并在屏幕上显示结果。

语法

```
mtoolstest
```

实例

在命令行中直接输入mtoolstest，即可显示mtools软件包当前的配置信息，结果如下：

```
$ mtoolstest #显示mtools 软件包当前的配置信息
drive J: #mtools软件包当前的配置信息列表
#fn=0 mode=0 builtin
file="/dev/sdb4" fat_bits=16
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0
mformat_only
drive Z:
#fn=0 mode=0 builtin
file="/dev/sdb4" fat_bits=16
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0
mformat_only
drive X:
#fn=0 mode=0 builtin
file="$DISPLAY" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0
drive A:
#fn=2 mode=128 defined in /etc/mtools.conf
file="/dev/fd0" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0
exclusive
drive B:
#fn=2 mode=128 defined in /etc/mtools.conf
file="/dev/fd1" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0
exclusive
drive M:
#fn=2 mode=0 defined in /etc/mtools.conf
file="/var/lib/dosemu/hdimage.first" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x80
partition=1
drive N:
#fn=2 mode=0 defined in /etc/mtools.conf
file="/var/lib/dosemu/fdimage" fat_bits=0
tracks=0 heads=0 sectors=0 hidden=0
offset=0x0
partition=0
mtools_fat_compatibility=0
mtools_skip_check=0
mtools_lower_case=0
```

Linux mv命令

Linux mv命令用来为文件或目录改名、或将文件或目录移入其它位置。

语法

```
mv [options] source dest
mv [options] source... directory
```

参数说明：

- -i: 若指定目录已有同名文件，则先询问是否覆盖旧文件;
- -f: 在mv操作要覆盖某已有的目标文件时不给任何指示;

mv参数设置与运行结果

命令格式	运行结果
mv 文件名 文件名	将源文件名改为目标文件名
mv 文件名 目录名	将文件移动到目标目录
mv 目录名 目录名	目标目录已存在，将源目录 移动到目标目录；目标 目录不存在则改名
mv 目录名 文件名	出错

实例

将文件 aaa 更名为 bbb :

```
mv aaa bbb
```

将info目录放入logs目录中。注意，如果logs目录不存在，则该命令将info改名为logs。

```
mv info/ logs
```

再如将/usr/student下的所有文件和目录移到当前目录下，命令行为：

```
$ mv /usr/student/*  .
```

Linux od命令

Linux od命令用于输出文件内容。

od指令会读取所给予的文件的内容，并将其内容以八进制字码呈现出来。

语法

```
od [-abcdfhilovx][-A <字码基数>][-j <字符数目>][-N <字符数目>][-s <字符串字符数>][-t <输出格式>]
```

参数：

- -a　此参数的效果和同时指定"-ta"参数相同。
- -A<字码基数>　选择要以何种基数计算字码。
- -b　此参数的效果和同时指定"-toC"参数相同。
- -c　此参数的效果和同时指定"-tC"参数相同。
- -d　此参数的效果和同时指定"-tu2"参数相同。
- -f　此参数的效果和同时指定"-tfF"参数相同。
- -h　此参数的效果和同时指定"-tx2"参数相同。
- -i　此参数的效果和同时指定"-td2"参数相同。
- -j<字符数目>或--skip-bytes=<字符数目>　略过设置的字符数目。
- -l　此参数的效果和同时指定"-td4"参数相同。
- -N<字符数目>或--read-bytes=<字符数目>　到设置的字符数目为止。
- -o　此参数的效果和同时指定"-to2"参数相同。
- -s<字符串字符数>或--strings=<字符串字符数>　只显示符合指定的字符数目的字符串。
- -t<输出格式>或--format=<输出格式>　设置输出格式。
- -v或--output-duplicates　输出时不省略重复的数据。
- -w<每列字符数>或--width=<每列字符数>　设置每列的最大字符数。
- -x　此参数的效果和同时指定"-h"参数相同。
- --help　在线帮助。
- --version　显示版本信息。

实例

创建 tmp 文件：

```
$ echo abcdef g > tmp
$ cat tmp
abcdef g
```

使用 od 命令：

```
$ od -b tmp
0000000 141 142 143 144 145 146 040 147 012
0000011
```

使用单字节八进制解释进行输出，注意左侧的默认地址格式为八字节:

```
$ od -c tmp
0000000   a   b   c   d   e   f       g  \n
0000011
```

使用ASCII码进行输出，注意其中包括转义字符

```
$ od -t d1 tmp
0000000   97   98   99  100  101  102   32  103   10
0000011
```

使用单字节十进制进行解释

```
$ od -A d -c tmp
0000000   a   b   c   d   e   f       g  \n
0000009
```

Linux paste命令

Linux paste命令用于合并文件的列。

paste指令会把每个文件以列对列的方式，一列列地加以合并。

语法

```
paste [-s][-d <间隔字符>][--help][--version][文件...]
```

参数：

- -d<间隔字符>或--delimiters=<间隔字符>　用指定的间隔字符取代跳格字符。
- -s或--serial　串列进行而非平行处理。
- --help　在线帮助。
- --version　显示帮助信息。
- [文件…] 指定操作的文件路径

实例

使用paste指令将文件"file"、"testfile"、"testfile1"进行合并，输入如下命令：

```
paste file testfile testfile1 #合并指定文件的内容
```

但是，在执行以上命令之前，首先使用"cat"指令对3个文件内容进行查看，显示如下所示：

```
$ cat file                  #file文件的内容
xiongdan 200
lihaihui 233
lymlrl 231
$ cat testfile              #testfile文件的内容
liangyuanm  ss
$ cat testfile1             #testfile1文件的内容
huanggai 56
zhixi 73
```

当合并指令"$ paste file testfile testfile1"执行后，程序界面中将显示合并后的文件内容，如下所示：

```
xiongdan 200
lihaihui 233
lymlrl 231
liangyuanm  ss
huanggai 56
zhixi 73
```

若使用paste指令的参数"-s"，则可以将一个文件中的多行数据合并为一行进行显示。例如，将文件"file"中的3行数据合并为一行数据进行显示，输入如下命令

```
$ paste -s file              #合并指定文件的多行数据
```

上面的命令执行后，显示的数据内容如下所示：

```
xiongdan 200 lihaihui 233 lymlrl 231
```

注意：参数"-s"只是将testfile文件的内容调整显示方式，并不会改变原文件的内容格式。

Linux patch命令

Linux patch命令用于修补文件。

patch指令让用户利用设置修补文件的方式，修改，更新原始文件。倘若一次仅修改一个文件，可直接在指令列中下达指令依序执行。如果配合修补文件的方式则能一次修补大批文件，这也是Linux系统核心的升级方法之一。

语法

```
patch [-bceEflnNRstTuvZ][-B <备份字首字符串>][-d <工作目录>][-D <标示符号>][-F <监别列数>][-g <
```

参数：

- -b或--backup 　备份每一个原始文件。
- -B<备份字首字符串>或--prefix=<备份字首字符串> 　设置文件备份时，附加在文件名称前面的字首字符串，该字符串可以是路径名称。
- -c或--context 　把修补数据解译成关联性的差异。
- -d<工作目录>或--directory=<工作目录> 　设置工作目录。
- -D<标示符号>或--ifdef=<标示符号> 　用指定的符号把改变的地方标示出来。
- -e或--ed 　把修补数据解译成ed指令可用的叙述文件。
- -E或--remove-empty-files 　若修补过后输出的文件其内容是一片空白，则移除该文件。
- -f或--force 　此参数的效果和指定"-t"参数类似，但会假设修补数据的版本为新　版本。
- -F<监别列数>或--fuzz<监别列数> 　设置监别列数的最大值。
- -g<控制数值>或--get=<控制数值> 　设置以RSC或SCCS控制修补作业。
- -i<修补文件>或--input=<修补文件> 　读取指定的修补问家你。
- -l或--ignore-whitespace 　忽略修补数据与输入数据的跳格，空格字符。
- -n或--normal 　把修补数据解译成一般性的差异。
- -N或--forward 　忽略修补的数据较原始文件的版本更旧，或该版本的修补数据已使　用过。
- -o<输出文件>或--output=<输出文件> 　设置输出文件的名称，修补过的文件会以该名称存放。
- -p<剥离层级>或--strip=<剥离层级> 　设置欲剥离几层路径名称。
- -f<拒绝文件>或--reject-file=<拒绝文件> 　设置保存拒绝修补相关信息的文件名称，预设的文件名称为.rej。
- -R或--reverse 　假设修补数据是由新旧文件交换位置而产生。
- -s或--quiet或--silent 　不显示指令执行过程，除非发生错误。
- -t或--batch 　自动略过错误，不询问任何问题。

- -T或--set-time 此参数的效果和指定"-Z"参数类似，但以本地时间为主。
- -u或--unified 把修补数据解译成一致化的差异。
- -v或--version 显示版本信息。
- -V<备份方式>或--version-control=<备份方式> 用"-b"参数备份目标文件后，备份文件的字尾会被加上一个备份字符串，这个字符串不仅可用"-z"参数变更，当使用"-V"参数指定不同备份方式时，也会产生不同字尾的备份字符串。
- -Y<备份字首字符串>或--basename-prefix=--<备份字首字符串> 设置文件备份时，附加在文件基本名称开头的字首字符串。
- -z<备份字尾字符串>或--suffix=<备份字尾字符串> 此参数的效果和指定"-B"参数类似，差别在于修补作业使用的路径与文件名若为src/linux/fs/super.c，加上"backup/"字符串后，文件super.c会备份于/src/linux/fs/backup目录里。
- -Z或--set-utc 把修补过的文件更改，存取时间设为UTC。
- --backup-if-mismatch 在修补数据不完全吻合，且没有刻意指定要备份文件时，才备份文件。
- --binary 以二进制模式读写数据，而不通过标准输出设备。
- --help 在线帮助。
- --nobackup-if-mismatch 在修补数据不完全吻合，且没有刻意指定要备份文件时，不要备份文件。
- --verbose 详细显示指令的执行过程。

实例

使用patch指令将文件"testfile1"升级，其升级补丁文件为"testfile.patch"，输入如下命令：

```
$ patch -p0 testfile1 testfile.patch    #使用补丁程序升级文件
```

使用该命令前，可以先使用指令"cat"查看"testfile1"的内容。在需要修改升级的文件与原文件之间使用指令"diff"比较可以生成补丁文件。具体操作如下所示：

```
$ cat testfile1                  #查看testfile1的内容
Hello,This is the firstfile!
$ cat testfile2                  #查看testfile2的内容
Hello,Thisisthesecondfile!
$ diff testfile1 testfile2           #比较两个文件
1c1
<Hello,Thisisthefirstfile!
---
>Hello,Thisisthesecondfile!
#将比较结果保存到tetsfile.patch文件
$ diff testfile1 testfile2>testfile.patch
$ cat testfile.patch                 #查看补丁包的内容
1c1
<Hello,Thisisthefirstfile!
---
>Hello,Thisisthesecondfile!
#使用补丁包升级testfile1文件
$ patch -p0 testfile1 testfile.patch
patching file testfile1
$cat testfile1                   #再次查看testfile1的内容
#testfile1文件被修改为与testfile2一样的内容
Hello,This is the secondfile!
```

注意：上述命令代码中，"$ diff testfile1 testfile2>testfile. patch"所使用的操作符"＞"表示将该操作符左边的文件数据写入到右边所指向的文件中。在这里，即是指将两个文件比较后的结果写入到文件"testfile.patch"中。

Linux rcp命令

Linux rcp命令用于复制远程文件或目录。

rcp指令用在远端复制文件或目录，如同时指定两个以上的文件或目录，且最后的目的地是一个已经存在的目录，则它灰把前面指定的所有文件或目录复制到该目录中。

语法

```
rcp [-pr][源文件或目录][目标文件或目录]
```

或

```
rcp [-pr][源文件或目录...][目标文件]
```

参数：

-p 保留源文件或目录的属性，包括拥有者，所属群组，权限与时间。

-r 递归处理，将指定目录下的文件与子目录一并处理。

实例

使用rcp指令复制远程文件到本地进行保存。

设本地主机当前账户为rootlocal，远程主机账户为root，要将远程主机（218.6.132.5）主目录下的文件"testfile"复制到本地目录"test"中，则输入如下命令：

```
rcp root@218.6.132.5:./testfile testfile  #复制远程文件到本地
rcp root@218.6.132.5:home/rootlocal/testfile testfile
#要求当前登录账户cmd 登录到远程主机
rcp 218.6.132.5:./testfile testfile
```

注意：指令"rcp"执行以后不会有返回信息，仅需要在目录"test"下查看是否存在文件"testfile"。若存在，则表示远程复制操作成功，否则远程复制操作失败。

Linux rm命令

Linux rm命令用于删除一个文件或者目录。

语法

```
rm [options] name...
```

参数：

- -i 删除前逐一询问确认。
- -f 即使原档案属性设为唯读，亦直接删除，无需逐一确认。
- -r 将目录及以下之档案亦逐一删除。

实例

删除文件可以直接使用rm命令，若删除目录则必须配合选项"-r"，例如：

```
# rm  test.txt
rm：是否删除 一般文件 "test.txt"? y
# rm  homework
rm: 无法删除目录"homework": 是一个目录
# rm  -r  homework
rm：是否删除 目录 "homework"? y
```

删除当前目录下的所有文件及目录，命令行为：

```
rm  -r  *
```

文件一旦通过rm命令删除，则无法恢复，所以必须格外小心地使用该命令。

Linux slocate命令

Linux slocate命令查找文件或目录。

slocate本身具有一个数据库，里面存放了系统中文件与目录的相关信息。

语法

```
slocate [-u][--help][--version][-d <目录>][查找的文件]
```

参数：

- -d<目录>或--database=<目录> 　指定数据库所在的目录。
- -u 　更新slocate数据库。
- --help 　显示帮助。
- --version 　显示版本信息。

实例

使用指令"slocate"显示文件名中含有关键字"fdisk"的文件路径信息，输入如下命令：

```
$ slocate fdisk #显示文件名中含有fdisk关键字的文件的路径信息
```

执行以上命令后，指令执行的输出信息如下：

```
$ slocate fdisk #显示文件名中含有fdisk 关键字的文件的路径信息
/root/cfdisk        #搜索到的文件路径列表
/root/fdisk
/root/sfdisk
/usr/include/grub/ieee1275/ofdisk.h
/usr/share/doc/util-Linux/README.cfdisk
/usr/share/doc/util-Linux/README.fdisk.gz
/usr/share/doc/util-Linux/examples/sfdisk.examples.gz
```

Linux split命令

Linux split命令用于将一个文件分割成数个。

该指令将大文件分割成较小的文件，在默认情况下将按照每1000行切割成一个小文件。

语法

```
split [--help][--version][-<行数>][-b <字节>][-C <字节>][-l <行数>][要切割的文件][输出文件名]
```

参数说明：

- -<行数> : 指定每多少行切成一个小文件
- -b<字节> : 指定每多少字节切成一个小文件
- --help : 在线帮助
- --version : 显示版本信息
- -C<字节> : 与参数"-b"相似，但是在切 割时将尽量维持每行的完整性
- [输出文件名] : 设置切割后文件的前置文件名， split会自动在前置文件名后再加上编号

实例

使用指令"split"将文件"README"每6行切割成一个文件，输入如下命令：

```
$ split -6 README       #将README文件每六行分割成一个文件
```

以上命令执行后，指令"split"会将原来的大文件"README"切割成多个以"x"开头的小文件。而在这些小文件中，每个文件都只有6行内容。

使用指令"ls"查看当前目录结构，如下所示：

```
$ ls                                #执行ls指令
#获得当前目录结构
README xaa xad xag xab xae xah xac xaf xai
```

Linux tee命令

Linux tee命令用于读取标准输入的数据，并将其内容输出成文件。

tee指令会从标准输入设备读取数据，将其内容输出到标准输出设备，同时保存成文件。

语法

```
tee [-ai][--help][--version][文件...]
```

参数：

- -a或--append 附加到既有文件的后面，而非覆盖它．
- -i或--ignore-interrupts 忽略中断信号。
- --help 在线帮助。
- --version 显示版本信息。

实例

使用指令"tee"将用户输入的数据同时保存到文件"file1"和"file2"中，输入如下命令：

```
$ tee file1 file2                   #在两个文件中复制内容
```

以上命令执行后，将提示用户输入需要保存到文件的数据，如下所示：

```
My Linux                        #提示用户输入数据
My Linux                        #输出数据，进行输出反馈
```

此时，可以分别打开文件"file1"和"file2"，查看其内容是否均是"My Linux"即可判断指令"tee"是否执行成功。

Linux tmpwatch命令

Linux tmpwatch命令用于删除暂存文件。

执行tmpwatch指令可删除不必要的暂存文件，您可以设置文件超期时间，单位以小时计算。

语法

```
tmpwatch [-afqv][--test][超期时间][目录...]
```

参数：

- -a或--all 　删除任何类型的文件。
- -f或--force 　强制删除文件或目录，其效果类似rm指令的"-f"参数。
- -q或--quiet 　不显示指令执行过程。
- -v或--verbose 　详细显示指令执行过程。
- -test 　仅作测试，并不真的删除文件或目录。

实例

使用指令"tmpwatch"删除目录"/tmp"中超过一天未使用的文件，输入如下命令：

```
$ tmpwatch 24 /tmp/ #删除/tmp目录中超过一天未使用的文件
```

以上命令执行后，其执行结果如下所示：

```
removing directctmp/orbit-tom if not empty
```

注意：该指令需要root权限，因此在使用tmpwatch命令前应该使用su命令切换用户。切换管理权限操作如下所示：

```
$ su                                #切换到root用户
口令：**********                    #输入用户密码
```

Linux touch命令

Linux touch命令用于修改文件或者目录的时间属性，包括存取时间和更改时间。若文件不存在，系统会建立一个新的文件。

ls -l 可以显示档案的时间记录。

语法

```
touch [-acfm][-d<日期时间>][-r<参考文件或目录>] [-t<日期时间>][--help][--version][文件或目录…]
```

- 参数说明：
- a 改变档案的读取时间记录。
- m 改变档案的修改时间记录。
- c 假如目的档案不存在，不会建立新的档案。与 --no-create 的效果一样。
- f 不使用，是为了与其他 unix 系统的相容性而保留。
- r 使用参考档的时间记录，与 --file 的效果一样。
- d 设定时间与日期，可以使用各种不同的格式。
- t 设定档案的时间记录，格式与 date 指令相同。
- --no-create 不会建立新档案。
- --help 列出指令格式。
- --version 列出版本讯息。

实例

使用指令"touch"修改文件"testfile"的时间属性为当前系统时间，输入如下命令：

```
$ touch testfile                #修改文件的时间属性
```

首先，使用ls命令查看testfile文件的属性，如下所示：

```
$ ls -l testfile                #查看文件的时间属性
#原来文件的修改时间为16:09
-rw-r--r-- 1 hdd hdd 55 2011-08-22 16:09 testfile
```

执行指令"touch"修改文件属性以后，并再次查看该文件的时间属性，如下所示：

```
$ touch testfile                #修改文件时间属性为当前系统时间
$ ls -l testfile                #查看文件的时间属性
#修改后文件的时间属性为当前系统时间
-rw-r--r-- 1 hdd hdd 55 2011-08-22 19:53 testfile
```

使用指令"touch"时，如果指定的文件不存在，则将创建一个新的空白文件。例如，在当前目录下，使用该指令创建一个空白文件"file"，输入如下命令：

```
$ touch file            #创建一个名为“file”的新的空白文件
```

Linux umask命令

Linux umask命令指定在建立文件时预设的权限掩码。

umask可用来设定[权限掩码]。[权限掩码]是由3个八进制的数字所组成，将现有的存取权限减掉权限掩码后，即可产生建立文件时预设的权限。

语法

```
umask [-S][权限掩码]
```

参数说明：

-S　以文字的方式来表示权限掩码。

实例

使用指令"umask"查看当前权限掩码，则输入下面的命令：

```
$ umask                         #获取当前权限掩码
```

执行上面的指令后，输出信息如下：

```
0022
```

接下来，使用指令"mkdir"创建一个目录，并使用指令"ls"获取该目录的详细信息，输入命令如下：

```
$ mkdir test1                       #创建目录
$ ls –d –l test1/                   #显示目录的详细信息
```

执行上面的命令后，将显示新创建目录的详细信息，如下所示：

```
drwxr-xr-x 2 rootlocal rootlocal 4096 2011-9-19 21:46 test1/
```

注意：在上面的输出信息中，"drwxr-xr-x"="777-022=755"。

Linux which命令

Linux which命令用于查找文件。

which指令会在环境变量$PATH设置的目录里查找符合条件的文件。

语法

```
which [文件...]
```

参数：

- -n<文件名长度> 指定文件名长度，指定的长度必须大于或等于所有文件中最长的文件名。
- -p<文件名长度> 与-n参数相同，但此处的<文件名长度>包括了文件的路径。
- -w 指定输出时栏位的宽度。
- -V 显示版本信息。

实例

使用指令"which"查看指令"bash"的绝对路径，输入如下命令：

```
$ which bash
```

上面的指令执行后，输出信息如下所示：

```
/bin/bash                      #bash可执行程序的绝对路径
```

Linux cp命令

Linux cp命令主要用于复制文件或目录。

语法

```
cp [options] source dest
```

或

```
cp [options] source... directory
```

参数说明：

- -a：此选项通常在复制目录时使用，它保留链接、文件属性，并复制目录下的所有内容。其作用等于dpR参数组合。
- -d：复制时保留链接。这里所说的链接相当于Windows系统中的快捷方式。
- -f：覆盖已经存在的目标文件而不给出提示。
- -i：与-f选项相反，在覆盖目标文件之前给出提示，要求用户确认是否覆盖，回答"y"时目标文件将被覆盖。
- -p：除复制文件的内容外，还把修改时间和访问权限也复制到新文件中。
- -r：若给出的源文件是一个目录文件，此时将复制该目录下所有的子目录和文件。
- -l：不复制文件，只是生成链接文件。

实例

使用指令"cp"将当前目录"test/"下的所有文件复制到新目录"newtest"下，输入如下命令：

```
$ cp –r test/ newtest
```

注意：用户使用该指令复制目录时，必须使用参数"-r"或者"-R"。

Linux mcopy命令

Linux mcopy命令用来复制 MSDOS 格式文件到 Linux 中，或是由 Linux 中复制 MSDOS 文件到磁片上。

mcopy 可复制单一的文件到所指定的文件名称，或是复制数个文件到所指定的目录之中。来源与目的文件可为 MSDOS 或是 Linux 文件。

mcopy指令是一种mtools工具指令，可以在DOS系统中复制文件或者在DOS与Linux操作系统之间进行文件复制。

语法

```
mcopy [-bnmpQt/][源文件][目标文件或目录]
```

参数：

- b 批处理模式。这是为大量的文件复制进行最佳化的选项,但是当在复制文件过程中产生 crash 时，会有安全性的问题产生。/ 递回的复制。包含目录所含文件与其下所有子目录中的文件。
- -n 覆盖其他文件时，不需要进行确认而直接覆盖
- m 将源文件修改时间设置为目标文件的修改时间。
- p 将源文件的属性设置为目标文件的属性。
- Q 当复制多个文件产生错误时，尽快结束程序。
- t 转换为文本文件。
- o 在覆盖 MSDOS 文件时不会出现警示讯息。

实例

将 A 盘根目录中的 autoexec.bat 复制到目前工作目录之下：

```
mcopy a:autoexec.bat .
```

当复制的内容包括子目录和文件时，必须使用参数"-/"递归操作，因此该命令为：

```
mcopy -/ A:\*
```

执行该命令前先使用mdir 命令查看原来的目录结构，执行mcopy 之后可使用ls 命令查看复制之后Linux系统中的文件结构，结果如下：

```
cmd@cmd-desktop:~$ mdir -/ a:\* #查看A 盘中的文件
Volume in drive A has no label #加载信息
Volume Serial Number is 13D2~055C
Directory for A:/ #以下为目录信息
#文件名目录大小 修改时间
./TEST <DIR> 2009-09-23 16:59
AUTORUN.INF 265 2009-09-23 16:53
AUTORUN.BAT 43 2009-09-23 16:56
3 files 308 bytes #统计总大小
724 325 bytes free #剩余空间
cmd@cmd-desktop:~$ mcopy -/ A:\* #将A盘上的所有文件复制到当前工作目录
cmd@cmd-desktop:~$ls
TEST AUTORUN.INF AUTORUN.BAT #A盘中的内容复制到Linux文件系统结构中
```

Linux mshowfat命令

Linux mshowfat命令用于显示MS-DOS文件在FAT中的记录。

mshowfat为mtools工具指令，可显示MS-DOS文件在FAT中的记录编号。

语法

```
mshowfat [文件...]
```

参数说明：

[文件…]： 执行操作的文件相对路径或者绝对路径

实例

使用指令mshowfat查看文件"autorun.bat"的FAT信息，输入如下命令：

```
$ mshowfat autorun.bat
```

以上命令执行后，文件"autorun.bat"的FAT相关信息将会被显示出来。

注意：执行操作的文件必须是DOS文件系统下的文件。

Linux rhmask命令

Linux rhmask命令用于对文件进行加密和解密操作。

执行rhmask指令可制作加密过的文件，方便用户在公开的网络上传输该文件，而不至于被任意盗用。

语法

```
rhmask [加密文件][输出文件] 或 rhmask [-d][加密文件][源文件][输出文件]
```

参数：

- -d　产生加密过的文件。

实例

使用指令"rhmask"将加密文件"code.txt"进行加密后，另存为输出文件"demo.txt"，输入如下命令：

```
$ rhmask code.txt demo.txt
```

以上命令执行后，文件"code.txt"将被加密后，另存为已经加密的文件"demo.txt"。

注意：该指令有两种语法，用户可以有选择性地进行使用即可。

Linux whereis命令

Linux whereis命令用于查找文件。

该指令会在特定目录中查找符合条件的文件。这些文件应属于原始代码、二进制文件，或是帮助文件。

该指令只能用于查找二进制文件、源代码文件和man手册页，一般文件的定位需使用locate命令。

语法

```
whereis [-bfmsu][-B <目录>...][-M <目录>...][-S <目录>...][文件...]
```

参数：

- -b 　只查找二进制文件。 *-B<目录>* 　只在设置的目录下查找二进制文件。 -f 　不显示文件名前的路径名称。 *-m* 　只查找说明文件。 -M<目录> 　只在设置的目录下查找说明文件。 *-s* 　只查找原始代码文件。 -S<目录> 　只在设置的目录下查找原始代码文件。* -u 　查找不包含指定类型的文件。

实例

使用指令"whereis"查看指令"bash"的位置，输入如下命令：

```
$ whereis bash
```

上面的指令执行后，输出信息如下所示：

```
bash:/bin/bash/etc/bash.bashrc/usr/share/man/man1/bash.1.gz
```

注意：以上输出信息从左至右分别为查询的程序名、bash路径、bash的man 手册页路径。

如果用户需要单独查询二进制文件或帮助文件，可使用如下命令：

```
$ whereis -b bash
$ whereis -m bash
```

输出信息如下：

```
$ whereis -b bash               #显示bash 命令的二进制程序
bash: /bin/bash /etc/bash.bashrc /usr/share/bash    # bash命令的二进制程序的地址
$ whereis -m bash               #显示bash 命令的帮助文件
bash: /usr/share/man/man1/bash.1.gz  #bash命令的帮助文件地址
```

Linux scp命令

Linux scp命令用于Linux之间复制文件和目录。

scp是 secure copy的缩写, scp是linux系统下基于ssh登陆进行安全的远程文件拷贝命令。

语法

```
scp [-1246BCpqrv] [-c cipher] [-F ssh_config] [-i identity_file]
[-l limit] [-o ssh_option] [-P port] [-S program]
[[user@]host1:]file1 [...] [[user@]host2:]file2
```

简易写法:

```
scp [可选参数] file_source file_target
```

参数说明：

- -1： 强制scp命令使用协议ssh1
- -2： 强制scp命令使用协议ssh2
- -4： 强制scp命令只使用IPv4寻址
- -6： 强制scp命令只使用IPv6寻址
- -B： 使用批处理模式（传输过程中不询问传输口令或短语）
- -C： 允许压缩。（将-C标志传递给ssh，从而打开压缩功能）
- -p：保留原文件的修改时间，访问时间和访问权限。
- -q： 不显示传输进度条。
- -r： 递归复制整个目录。
- -v：详细方式显示输出。scp和ssh(1)会显示出整个过程的调试信息。这些信息用于调试连接，验证和配置问题。
- -c cipher： 以cipher将数据传输进行加密，这个选项将直接传递给ssh。
- -F ssh_config： 指定一个替代的ssh配置文件，此参数直接传递给ssh。
- -i identity_file： 从指定文件中读取传输时使用的密钥文件，此参数直接传递给ssh。
- -l limit： 限定用户所能使用的带宽，以Kbit/s为单位。
- -o ssh_option： 如果习惯于使用ssh_config(5)中的参数传递方式，
- -P port：注意是大写的P, port是指定数据传输用到的端口号
- -S program： 指定加密传输时所使用的程序。此程序必须能够理解ssh(1)的选项。

实例

1、从本地复制到远程

命令格式：

```
scp local_file remote_username@remote_ip:remote_folder
或者
scp local_file remote_username@remote_ip:remote_file
或者
scp local_file remote_ip:remote_folder
或者
scp local_file remote_ip:remote_file
```

- 第1,2个指定了用户名，命令执行后需要再输入密码，第1个仅指定了远程的目录，文件名字不变，第2个指定了文件名；
- 第3,4个没有指定用户名，命令执行后需要输入用户名和密码，第3个仅指定了远程的目录，文件名字不变，第4个指定了文件名；

应用实例：

```
scp /home/space/music/1.mp3 root@www.w3cschool.cc:/home/root/others/music
scp /home/space/music/1.mp3 root@www.w3cschool.cc:/home/root/others/music/001.mp3
scp /home/space/music/1.mp3 www.w3cschool.cc:/home/root/others/music
scp /home/space/music/1.mp3 www.w3cschool.cc:/home/root/others/music/001.mp3
```

复制目录命令格式：

```
scp -r local_folder remote_username@remote_ip:remote_folder
或者
scp -r local_folder remote_ip:remote_folder
```

- 第1个指定了用户名，命令执行后需要再输入密码；
- 第2个没有指定用户名，命令执行后需要输入用户名和密码；

应用实例：

```
scp -r /home/space/music/ root@www.w3cschool.cc:/home/root/others/
scp -r /home/space/music/ www.w3cschool.cc:/home/root/others/
```

上面命令将本地 music 目录复制到远程 others 目录下。

2、从远程复制到本地

从远程复制到本地，只要将从本地复制到远程的命令的后2个参数调换顺序即可，如下实例

应用实例：

```
scp root@www.w3cschool.cc:/home/root/others/music /home/space/music/1.mp3
scp -r www.w3cschool.cc:/home/root/others/ /home/space/music/
```

说明

1.如果远程服务器防火墙有为scp命令设置了指定的端口，我们需要使用 -p 参数来设置命令的端口号，命令格式如下：

```
#scp命令使用端口号 4588
scp -p 4588 remote@www.w3cschool.cc:/usr/local/sin.sh /home/administrator
```

2.使用scp命令要确保使用的用户具有可读取远程服务器相应文件的权限，否则scp命令是无法起作用的。

Linux awk 命令

AWK是一种处理文本文件的语言，是一个强大的文本分析工具。

之所以叫AWK是因为其取了三位创始人 Alfred Aho，Peter Weinberger, 和 Brian Kernighan 的Family Name的首字符。

语法

```
awk [选项参数] 'script' var=value file(s)
或
awk [选项参数] -f scriptfile var=value file(s)
```

选项参数说明：

- -F fs or --field-separator fs
  指定输入文件折分隔符，fs是一个字符串或者是一个正则表达式，如-F:。
- -v var=value or --asign var=value
  赋值一个用户定义变量。
- -f scripfile or --file scriptfile
  从脚本文件中读取awk命令。
- -mf nnn and -mr nnn
  对nnn值设置内在限制，-mf选项限制分配给nnn的最大块数目；-mr选项限制记录的最大数目。这两个功能是Bell实验室版awk的扩展功能，在标准awk中不适用。
- -W compact or --compat, -W traditional or --traditional
  在兼容模式下运行awk。所以gawk的行为和标准的awk完全一样，所有的awk扩展都被忽略。
- -W copyleft or --copyleft, -W copyright or --copyright
  打印简短的版权信息。
- -W help or --help, -W usage or --usage
  打印全部awk选项和每个选项的简短说明。
- -W lint or --lint
  打印不能向传统unix平台移植的结构的警告。
- -W lint-old or --lint-old
  打印关于不能向传统unix平台移植的结构的警告。
- -W posix
  打开兼容模式。但有以下限制，不识别：/x、函数关键字、func、换码序列以及当fs是一个空格时，将新行作为一个域分隔符；操作符和=不能代替^和^=；fflush无效。
- -W re-interval or --re-inerval
  允许间隔正则表达式的使用，参考(grep中的Posix字符类)，如括号表达式[[:alpha:]]。

- -W source program-text or --source program-text
  使用program-text作为源代码，可与-f命令混用。
- -W version or --version
  打印bug报告信息的版本。

基本用法

log.txt文本内容如下：

```
2 this is a test
3 Are you like awk
This's a test
10 There are orange,apple,mongo
```

用法一：

```
awk '{[pattern] action}' {filenames}   # 行匹配语句 awk '' 只能用单引号
```

实例：

```
# 每行按空格或TAB分割，输出文本中的1、4项
 $ awk '{print $1,$4}' log.txt
 ---------------------------------------------
 2 a
 3 like
 This's
 10 orange,apple,mongo
 # 格式化输出
 $ awk '{printf "%-8s %-10s\n",$1,$4}' log.txt
 ---------------------------------------------
 2        a
 3        like
 This's
 10       orange,apple,mongo
```

用法二：

```
awk -F  #-F相当于内置变量FS, 指定分割字符
```

实例：

```
# 使用","分割
 $  awk -F, '{print $1,$2}'   log.txt
 ---------------------------------------------
 2 this is a test
 3 Are you like awk
 This's a test
 10 There are orange apple
 # 或者使用内建变量
 $ awk 'BEGIN{FS=","} {print $1,$2}'     log.txt
 ---------------------------------------------
 2 this is a test
 3 Are you like awk
 This's a test
 10 There are orange apple
 # 使用多个分隔符.先使用空格分割，然后对分割结果再使用","分割
 $ awk -F '[ ,]'  '{print $1,$2,$5}'   log.txt
 ---------------------------------------------
 2 this test
 3 Are awk
 This's a
 10 There apple
```

用法三：

```
awk -v  # 设置变量
```

实例：

```
 $ awk -va=1 '{print $1,$1+a}' log.txt
 ---------------------------------------------
 2 3
 3 4
 This's 1
 10 11
 $ awk -va=1 -vb=s '{print $1,$1+a,$1b}' log.txt
 ---------------------------------------------
 2 3 2s
 3 4 3s
 This's 1 This'ss
 10 11 10s
```

用法四：

```
awk -f {awk脚本} {文件名}
```

实例：

```
 $ awk -f cal.awk log.txt
```

运算符

运算符	描述
= += -= = /= %= ^= *=	赋值
?:	C条件表达式
||	逻辑或
&&	逻辑与
~ ~!	匹配正则表达式和不匹配正则表达式
< <= > >= != ==	关系运算符
空格	连接
+ -	加，减
* / &	乘，除与求余
+ - !	一元加，减和逻辑非
^ *	求幂
++ --	增加或减少，作为前缀或后缀
$	字段引用
in	数组成员

过滤第一列大于2的行

```
$ awk '$1>2' log.txt    #命令
#输出
3 Are you like awk
This's a test
10 There are orange,apple,mongo
```

过滤第一列等于2的行

```
$ awk '$1==2 {print $1,$3}' log.txt    #命令
#输出
2 is
```

过滤第一列大于2并且第二列等于'Are'的行

```
$ awk '$1>2 && $2=="Are" {print $1,$2,$3}' log.txt    #命令
#输出
3 Are you
```

内建变量

变量	描述
$n	当前记录的第n个字段，字段间由FS分隔
$0	完整的输入记录
ARGC	命令行参数的数目
ARGIND	命令行中当前文件的位置(从0开始算)
ARGV	包含命令行参数的数组
CONVFMT	数字转换格式(默认值为%.6g)ENVIRON环境变量关联数组
ERRNO	最后一个系统错误的描述
FIELDWIDTHS	字段宽度列表(用空格键分隔)
FILENAME	当前文件名
FNR	同NR，但相对于当前文件
FS	字段分隔符(默认是任何空格)
IGNORECASE	如果为真，则进行忽略大小写的匹配
NF	当前记录中的字段数
NR	当前记录数
OFMT	数字的输出格式(默认值是%.6g)
OFS	输出字段分隔符(默认值是一个空格)
ORS	输出记录分隔符(默认值是一个换行符)
RLENGTH	由match函数所匹配的字符串的长度
RS	记录分隔符(默认是一个换行符)
RSTART	由match函数所匹配的字符串的第一个位置
SUBSEP	数组下标分隔符(默认值是/034)

```
$ awk 'BEGIN{printf "%4s %4s %4s %4s %4s %4s %4s %4s %4s\n","FILENAME","ARGC","FNR","FS",
FILENAME ARGC  FNR   FS   NF   NR  OFS  ORS   RS
---------------------------------------------
log.txt    2    1         5    1
log.txt    2    2         5    2
log.txt    2    3         3    3
log.txt    2    4         4    4
$ awk -F\' 'BEGIN{printf "%4s %4s %4s %4s %4s %4s %4s %4s %4s\n","FILENAME","ARGC","FNR",
FILENAME ARGC  FNR   FS   NF   NR  OFS  ORS   RS
---------------------------------------------
log.txt    2    1    '    1    1
log.txt    2    2    '    1    2
log.txt    2    3    '    2    3
log.txt    2    4    '    1    4
# 输出顺序号 NR, 匹配文本行号
$ awk '{print NR,FNR,$1,$2,$3}' log.txt
---------------------------------------------
1 1 2 this is
2 2 3 Are you
3 3 This's a test
4 4 10 There are
# 指定输出分割符
$  awk '{print $1,$2,$5}' OFS=" $ "  log.txt
---------------------------------------------
2 $ this $ test
3 $ Are $ awk
This's $ a $
10 $ There $
```

使用正则，字符串匹配

```
# 输出第二列包含 "th"，并打印第二列与第四列
$ awk '$2 ~ /th/ {print $2,$4}' log.txt
---------------------------------------------
this a
```

~ 表示模式开始。// 中是模式。

```
# 输出包含"re" 的行
$ awk '/re/ ' log.txt
---------------------------------------------
3 Are you like awk
10 There are orange,apple,mongo
```

忽略大小写

```
$ awk 'BEGIN{IGNORECASE=1} /this/' log.txt
---------------------------------------------
2 this is a test
This's a test
```

模式取反

```
$ awk '$2 !~ /th/ {print $2,$4}' log.txt
---------------------------------------------
Are like
a
There orange,apple,mongo
$ awk '!/th/ {print $2,$4}' log.txt
---------------------------------------------
Are like
a
There orange,apple,mongo
```

awk脚本

关于awk脚本，我们需要注意两个关键词BEGIN和END。

- BEGIN{ 这里面放的是执行前的语句 }
- END {这里面放的是处理完所有的行后要执行的语句 }
- {这里面放的是处理每一行时要执行的语句}

假设有这么一个文件（学生成绩表）：

```
$ cat score.txt
Marry   2143 78 84 77
Jack    2321 66 78 45
Tom     2122 48 77 71
Mike    2537 87 97 95
Bob     2415 40 57 62
```

我们的awk脚本如下：

```
$ cat cal.awk
#!/bin/awk -f
#运行前
BEGIN {
    math = 0
    english = 0
    computer = 0

    printf "NAME    NO.   MATH  ENGLISH  COMPUTER   TOTAL\n"
    printf "---------------------------------------------\n"
}
#运行中
{
    math+=$3
    english+=$4
    computer+=$5
    printf "%-6s %-6s %4d %8d %8d %8d\n", $1, $2, $3,$4,$5, $3+$4+$5
}
#运行后
END {
    printf "---------------------------------------------\n"
    printf "  TOTAL:%10d %8d %8d \n", math, english, computer
    printf "AVERAGE:%10.2f %8.2f %8.2f\n", math/NR, english/NR, computer/NR
}
```

我们来看一下执行结果：

```
$ awk -f cal.awk score.txt
NAME    NO.   MATH  ENGLISH  COMPUTER   TOTAL
---------------------------------------------
Marry  2143     78       84       77      239
Jack   2321     66       78       45      189
Tom    2122     48       77       71      196
Mike   2537     87       97       95      279
Bob    2415     40       57       62      159
---------------------------------------------
  TOTAL:       319      393      350
AVERAGE:     63.80    78.60    70.00
```

另外一些实例

AWK的hello world程序为：

```
BEGIN { print "Hello, world!" }
```

计算文件大小

```
$ ls -l *.txt | awk '{sum+=$6} END {print sum}'
--------------------------------------------------
666581
```

从文件中找出长度大于80的行

```
awk 'lenght>80' log.txt
```

打印九九乘法表

```
seq 9 | sed 'H;g' | awk -v RS='' '{for(i=1;i<=NF;i++)printf("%dx%d=%d%s", i, NR, i*NR, i=
```

更多详细内容可以查看 AWK 官方手册：http://www.gnu.org/software/gawk/manual/gawk.html

Linux命令大全 - 文档编辑

col	colrm	comm	csplit
ed	egrep	ex	fgrep
fmt	fold	grep	ispell
jed	joe	join	look
mtype	pico	rgrep	sed
sort	spell	tr	expr
uniq	wc		

Linux col命令

Linux col命令用于过滤控制字符。

在许多UNIX说明文件里，都有RLF控制字符。当我们运用shell特殊字符">"和">>"，把说明文件的内容输出成纯文本文件时，控制字符会变成乱码，col指令则能有效滤除这些控制字符。

语法

```
col [-bfx][-l<缓冲区列数>]
```

参数：

- -b 过滤掉所有的控制字符，包括RLF和HRLF。
- -f 滤除RLF字符，但允许将HRLF字符呈现出来。
- -x 以多个空格字符来表示跳格字符。
- -l<缓冲区列数> 预设的内存缓冲区有128列，您可以自行指定缓冲区的大小。

实例

下面以 man 命令帮助文档为例，讲解col 命令的使用。

将man 命令的帮助文档保存为man_help，使用-b 参数过滤所有控制字符。在终端中使用如下命令：

```
man man | col-b > man_help
```

注:其中"|"用于建立管道，把man命令的输出结果转为col命令的输入数据。

Linux colrm命令

Linux colrm命令用于滤掉指定的行。

colrm指令从标准输入设备读取书记，转而输出到标准输出设备。如果不加任何参数，则该指令不会过滤任何一行。

语法

```
colrm [开始行数编号<结束行数编号>]</p>

<p><b>参数说明：</b></p>
<ul>
<li>开始行数编号： 指定要删除的列的起始编号。</li>
<li>结束行数编号： 指定要删除的列的结束编号，有时候这个参数可以省略。</li>
</ul>
<h3>实例</h3>
<p>不带任何参数时该命令不会删除任何列：</p>
<pre>
colrm
```

按回车键后，光标将在第一行闪烁，等待标准输入，此时输入字符，如"Hello Linux！"，再按回车键后第二行将出现与第一行相同内容，此时按Ctrl+C组合键可以退出。终端中显示的内容如下所示：

```
cmd@hdd-desktop:~$ colrm
Hello Linux! #输入Hello Linux！字符串
Hello Linux! #输出刚才输入的字符串Hello Linux！
```

如想要删除第4 列之后的所有内容，可以使用如下命令：

```
colrm 4
```

类似于上例，此时标准输入等待输入，用户输入字符串按回车键后，将输出如下结果：

```
cmd@hdd-desktop:~$ colrm 4
Hello Linux! #输入Hello Linux！字符串
Hel #输出删除了第4列以后所有内容的字符串
```

删除指定列的内容。如删除第4列到第6列的内容，可使用如下命令：

```
colrm 4 6
```

输出的结果如下：

```
cmd@hdd-desktop:~$ colrm 4 6
Hello Linux! #输入Hello Linux！字符串
HelLinux! #输出删除了从第4列到第6列字符的字符串
```

Linux comm命令

Linux comm命令用于比较两个已排过序的文件。

这项指令会一列列地比较两个已排序文件的差异，并将其结果显示出来，如果没有指定任何参数，则会把结果分成3行显示：第1行仅是在第1个文件中出现过的列，第2行是仅在第2个文件中出现过的列，第3行则是在第1与第2个文件里都出现过的列。若给予的文件名称为"-"，则comm指令会从标准输入设备读取数据。

语法

```
comm [-123][--help][--version][第1个文件][第2个文件]
```

参数：

- -1 不显示只在第1个文件里出现过的列。
- -2 不显示只在第2个文件里出现过的列。
- -3 不显示只在第1和第2个文件里出现过的列。
- --help 在线帮助。
- --version 显示版本信息。

实例

aaa.txt 与 bbb.txt 的文件内容如下：

```
[root@localhost text]# cat aaa.txt
aaa
bbb
ccc
ddd
eee
111
222
[root@localhost text]# cat bbb.txt
bbb
ccc
aaa
hhh
ttt
jjj
```

```
<p>执行 comm 命令输出结果如下：</p>
[root@localhost text]# comm aaa.txt bbb.txt
aaa
                bbb
                ccc
        aaa
ddd
eee
111
222
        hhh
        ttt
        jjj
第一列  第二列  第三列
```

输出的第一列只包含在aaa.txt中出现的行，第二列包含在bbb.txt中出现的行，第三列包含在aaa.txt和bbb.txt中相同的行。各列是以制表符（\t）作为定界符。

Linux csplit命令

Linux csplit命令用于分割文件。

将文件依照指定的范本样式予以切割后，分别保存成名称为xx00,xx01,xx02...的文件。若给予的文件名称为"-"，则csplit指令会从标准输入设备读取数据。

语法

```
csplit [-kqsz][-b<输出格式>][-f<输出字首字符串>]
[-n<输出文件名位数>][--help][--version][文件][范本样式...]
```

参数：

- -b<输出格式>或--suffix-format=<输出格式> 预设的输出格式其文件名称为xx00,xx01...等，您可以通过改变<输出格式>来改变输出的文件名。
- -f<输出字首字符串>或--prefix=<输出字首字符串> 预设的输出字首字符串其文件名为xx00,xx01...等，如果你指定输出字首字符串为"hello"，则输出的文件名称会变成hello00,hello01...等。
- -k或--keep-files 保留文件，就算发生错误或中断执行，也不能删除已经输出保存的文件。
- -n<输出文件名位数>或--digits=<输出文件名位数> 预设的输出文件名位数其文件名称为xx00,xx01...等，如果你指定输出文件名位数为"3"，则输出的文件名称会变成xx000,xx001...等。
- -q或-s或--quiet或--silent 不显示指令执行过程。
- -z或--elide-empty-files 删除长度为0 Byte文件。
- --help 在线帮助。
- --version 显示版本信息。

实例

将文本文件testfile以第 2 行为分界点切割成两份，使用如下命令：

```
csplit testfile 2
```

testfile文件中的内容如下：

```
$ cat testfile #查看testfile 文件内容
hello Linux!
Linux is a free Unix-type operating system.
This is a Linux testfile!
Linux
```

使用csplit命令，输出结果如下：

```
$ csplit testfile 2
13 #xx00文件字符个数
76 #xx01文件字符个数
```

其中第1 行是第一个文件xx00的字符个数，同样，第2 行为第二个文件xx01的字符个数。同时，在testfile 的同目录下将生成两个文件，文件名分别为xx00、xx01，xx00 中的内容为：

```
$ cat xx00 #查看分割后的xx00文件内容
hello Linux! #testfile文件第1行的内容
```

xx01 中的内容为：

```
$ cat xx01 #查看分割后的xx01文件内容
Linux is a free Unix-type operating system. #testfile文件第2行以后的内容
This is a Linux testfile!
Linux
```

Linux ed命令

Linux ed命令是文本编辑器，用于文本编辑。

ed是Linux中功能最简单的文本编辑程序，一次仅能编辑一行而非全屏幕方式的操作。

ed命令并不是一个常用的命令，一般使用比较多的是vi 指令。但ed文本编辑器对于编辑大文件或对于在shell脚本程序中进行文本编辑很有用。

语法

```
ed [-][-Gs][-p<字符串>][--help][--version][文件]
```

参数：

- -G或--traditional 提供回兼容的功能。
- -p<字符串> 指定ed在command mode的提示字符。
- -s,-,--quiet或--silent 不执行开启文件时的检查功能。
- --help 显示帮助。
- --version 显示版本信息。

实例

以下是一个 Linux ed 完整实例解析：

```
$ ed                <- 激活 ed 命令
a                   <- 告诉 ed 我要编辑新文件
My name is Titan. <- 输入第一行内容
And I love Perl very much. <- 输入第二行内容
.                   <- 返回 ed 的命令行状态
i                   <- 告诉 ed 我要在最后一行之前插入内容
I am 24\.            <- 将“I am 24.”插入“My name is Titan.”和“And I love Perl very much.”之间
.                   <- 返回 ed 的命令行状态
c                   <- 告诉 ed 我要替换最后一行输入内容
I am 24 years old. <- 将“I am 24.”替换成“I am 24 years old.”（注意：这里替换的是最后输的内容）
.                   <- 返回 ed 的命令行状态
w readme.text       <- 将文件命名为“readme.text”并保存（注意：如果是编辑已经存在的文件，只需要敲入 w .
q                   <- 完全退出 ed 编辑器
```

这是文件的内容是：

```
$ cat readme.text
My name is Titan.
I am 24 years old.
And I love Perl vrey much.
```

Linux egrep命令

Linux egrep命令用于在文件内查找指定的字符串。

egrep执行效果与"grep-E"相似，使用的语法及参数可参照grep指令，与grep的不同点在于解读字符串的方法。

egrep是用extended regular expression语法来解读的，而grep则用basic regular expression语法解读，extended regular expression比basic regular expression的表达更规范。

语法

```
egrep [范本模式] [文件或目录]
```

参数说明：

- [范本模式] ：查找的字符串规则。
- [文件或目录] ：查找的目标文件或目录。

实例

显示文件中符合条件的字符。例如，查找当前目录下所有文件中包含字符串"Linux"的文件，可以使用如下命令：

```
egrep Linux *
```

结果如下所示：

```
$ egrep Linux * #查找当前目录下包含字符串“Linux”的文件
testfile:hello Linux! #以下五行为testfile 中包含Linux字符的行
testfile:Linux is a free Unix-type operating system.
testfile:This is a Linux testfile!
testfile:Linux
testfile:Linux
testfile1:helLinux! #以下两行为testfile1中含Linux字符的行
testfile1:This a Linux testfile!
#以下两行为testfile_2 中包含Linux字符的行
testfile_2:Linux is a free unix-type opterating system.
testfile_2:Linux test
xx00:hello Linux! #xx00包含Linux字符的行
xx01:Linux is a free Unix-type operating system. #以下三行为xx01包含Linux字符的行
xx01:This is a Linux testfile!
xx01:Linux
```

Linux ex命令

Linux ex命令用于在Ex模式下启动vim文本编辑器。

ex执行效果如同vi -E，使用语法及参数可参照vi指令，如要从Ex模式回到普通模式，则在vim中输入":vi"或":visual"指令即可。

语法

```
ex [选项][参数]
```

参数说明：

- -b：使用二进制模式编辑文件
- -c 指令：编辑完第一个文件后执行指定的指令
- -d ：编辑多个文件时，显示差异部分
- -m ：不允许修改文件
- -n ：不使用缓存
- -oN：其中 N 为数字
- -r ：列出缓存，并显示恢复信息
- -R ：以只读的方式打开文件
- -s ：不显示任何错误信息
- -V ：显示指令的详细执行过程
- --help ：显示帮助信息
- --version ：显示版本信息

实例

在ex 指令后输入文件名按回车键后，即可进入ex 编辑模式，如编辑testfile文件，使用的命令格式如下：

```
ex testfile
```

输出的信息如下：

```
"testfile" 5L, 95C
```

"testfile"表示文件名，5L表示5 行，95 表示字节数

进入ex 模式。输入"visual"回到正常模式

它的操作与vim 中是一样的，此时如果在":"后输入"visual"后按回车键，将进入到vi 指令全屏界面；如果输入"q"，则退出编辑器。

Linux fgrep命令

本指令相当于执行grep指令加上参数"-F"，详见grep命令说明。

Linux fgrep命令用于查找文件里符合条件的字符串。

语法

```
fgrep [范本样式][文件或目录...]
```

实例

具体使用实例请参考grep命令。

Linux fmt命令

Linux fmt命令用于编排文本文件。

fmt指令会从指定的文件里读取内容，将其依照指定格式重新编排后，输出到标准输出设备。若指定的文件名为"-"，则fmt指令会从标准输入设备读取数据。

语法

```
fmt [-cstu][-p<列起始字符串>][-w<每列字符数>][--help][--version][文件...]
```

参数说明：

- -c或--crown-margin 每段前两列缩排。
- -p<列起始字符串>或-prefix=<列起始字符串> 仅合并含有指定字符串的列，通常运用在程序语言的注解方面。
- -s或--split-only 只拆开字数超出每列字符数的列，但不合并字数不足每列字符数的列。
- -t或--tagged-paragraph 每列前两列缩排，但第1列和第2列的缩排格式不同。
- -u或--uniform-spacing 每个字符之间都以一个空格字符间隔，每个句子之间则两个空格字符分隔。
- -w<每列字符数>或--width=<每列字符数>或-<每列字符数> 设置每列的最大字符数。
- --help 在线帮助。
- --version 显示版本信息。

实例

重排指定文件。如文件testfile共5 行文字，可以通过命令对该文件格式进行重排，其命令为：

```
fmt testfile
```

输出结果如下：

```
$ fmt testfile #重排testfile 文件
hello Linux! Linux is a free Unix-type operating system. This is a
Linux testfile! Linux Linux
```

将文件testfile重新排成85 个字符一行，并在标准输出设备上输出，其命令应该为：

```
fmt -w 85 testfile
```

为了对比，先使用cat 命令查看文件内容：

```
$ cat testfile #查看testfile 文件的内容
hello Linux!
Linux is a free Unix-type operating system.
This is a Linux testfile!
Linux
Linux
```

使用fmt命令重排之后，输出结果如下：

```
$ fmt -w 85 testfile #指定重排宽度为85个字符
hello Linux! Linux is a free Unix-type operating system. This is a Linux testfile!
Linux Linux
```

Linux fold命令

Linux fold命令用于限制文件列宽。

fold指令会从指定的文件里读取内容，将超过限定列宽的列加入增列字符后，输出到标准输出设备。若不指定任何文件名称，或是所给予的文件名为"-"，则fold指令会从标准输入设备读取数据。

语法

```
fold [-bs][-w<每列行数>][--help][--version][文件...]
```

参数：

- -b或--bytes 以Byte为单位计算列宽，而非采用行数编号为单位。
- -s或--spaces 以空格字符作为换列点。
- -w<每列行数>或--width<每列行数> 设置每列的最大行数。
- --help 在线帮助。
- --version 显示版本信息。

实例

将一个名为testfile 的文件的行折叠成宽度为30，可使用如下命令：

```
fold -w 30 testfile
```

为了对比，先将testfile文件输出如下：

```
$ cat testfile #查看testfile 中的内容
Linux networks are becoming more and more common, but
security is often an overlooked
issue. Unfortunately, in today’s environment all networks
are potential hacker targets,
from top-secret military research networks to small home LANs.
Linux Network Security focuses on securing Linux in a
networked environment, where the
security of the entire network needs to be considered
rather than just isolated machines.
It uses a mix of theory and practical techniques to
teach administrators how to install and
use security applications, as well as how the
applications work and why they are necessary.
```

然后使用fold命令折叠显示：

```
$ fold -w 30 testfile #行折叠成宽度为30，显示testfile 文件
Linux networks are becoming mo
re and more common, but securi
ty is often an overlooked issu
e. Unfortunately, in today’s
environment all networks are
potential hacker targets, from
top-secret military research
networks to small home LANs.
Linux Network Security focuses
on securing Linux in a networ
ked environment, where the sec
urity of the entire network ne
eds to be considered rather th
an just isolated machines. It
uses a mix of theory and pract
ical techniques to teach admin
istrators how to install and u
se security applications, as w
ell as how the applications wo
rk and why they are necessary
```

Linux grep命令

Linux grep命令用于查找文件里符合条件的字符串。

grep指令用于查找内容包含指定的范本样式的文件，如果发现某文件的内容符合所指定的范本样式，预设grep指令会把含有范本样式的那一列显示出来。若不指定任何文件名称，或是所给予的文件名为"-"，则grep指令会从标准输入设备读取数据。

语法

```
grep [-abcEFGhHilLnqrsvVwxy][-A<显示列数>][-B<显示列数>][-C<显示列数>][-d<进行动作>][-e<范本样式
```

参数：

- -a或--text 不要忽略二进制的数据。
- -A<显示列数>或--after-context=<显示列数> 除了显示符合范本样式的那一列之外，并显示该列之后的内容。
- -b或--byte-offset 在显示符合范本样式的那一列之前，标示出该列第一个字符的位编号。
- -B<显示列数>或--before-context=<显示列数> 除了显示符合范本样式的那一列之外，并显示该列之前的内容。
- -c或--count 计算符合范本样式的列数。
- -C<显示列数>或--context=<显示列数>或-<显示列数> 除了显示符合范本样式的那一列之外，并显示该列之前后的内容。
- -d<进行动作>或--directories=<进行动作> 当指定要查找的是目录而非文件时，必须使用这项参数，否则grep指令将回报信息并停止动作。
- -e<范本样式>或--regexp=<范本样式> 指定字符串做为查找文件内容的范本样式。
- -E或--extended-regexp 将范本样式为延伸的普通表示法来使用。
- -f<范本文件>或--file=<范本文件> 指定范本文件，其内容含有一个或多个范本样式，让grep查找符合范本条件的文件内容，格式为每列一个范本样式。
- -F或--fixed-regexp 将范本样式视为固定字符串的列表。
- -G或--basic-regexp 将范本样式视为普通的表示法来使用。
- -h或--no-filename 在显示符合范本样式的那一列之前，不标示该列所属的文件名称。
- -H或--with-filename 在显示符合范本样式的那一列之前，表示该列所属的文件名称。
- -i或--ignore-case 忽略字符大小写的差别。
- -l或--file-with-matches 列出文件内容符合指定的范本样式的文件名称。
- -L或--files-without-match 列出文件内容不符合指定的范本样式的文件名称。
- -n或--line-number 在显示符合范本样式的那一列之前，标示出该列的列数编号。
- -q或--quiet或--silent 不显示任何信息。

- -r或--recursive 此参数的效果和指定"-d recurse"参数相同。
- -s或--no-messages 不显示错误信息。
- -v或--revert-match 反转查找。
- -V或--version 显示版本信息。
- -w或--word-regexp 只显示全字符合的列。
- -x或--line-regexp 只显示全列符合的列。
- -y 此参数的效果和指定"-i"参数相同。
- --help 在线帮助。

实例

1、在当前目录中，查找后缀有"test"字样的文件中包含"test"字符串的文件，并打印出该字符串的行。此时，可以使用如下命令：

```
grep test *file
```

结果如下所示：

```
$ grep test test* #查找后缀有“test”的文件包含“test”字符串的文件
testfile1:This a Linux testfile! #列出testfile1 文件中包含test字符的行
testfile_2:This is a linux testfile! #列出testfile_2 文件中包含test字符的行
testfile_2:Linux test #列出testfile_2 文件中包含test字符的行
```

2、以递归的方式查找符合条件的文件。例如，查找指定目录/etc/acpi 及其子目录（如果存在子目录的话）下所有文件中包含字符串"update"的文件，并打印出该字符串所在行的内容，使用的命令为：

```
grep -r update /etc/acpi
```

输出结果如下：

```
$ grep-r update /etc/acpi #以递归的方式查找“etc/acpi”
#下包含“update”的文件
/etc/acpi/ac.d/85-anacron.sh:# (Things like the slocate updatedb cause a lot of IO.)
Rather than
/etc/acpi/resume.d/85-anacron.sh:# (Things like the slocate updatedb cause a lot of
IO.) Rather than
/etc/acpi/events/thinkpad-cmos:action=/usr/sbin/thinkpad-keys--update
```

3、反向查找。前面各个例子是查找并打印出符合条件的行，通过"-v"参数可以打印出不符合条件行的内容。

查找文件名中包含test 的文件中不包含test 的行，此时，使用的命令为：

```
grep -v test*
```

结果如下所示：

```
$ grep-v test* #查找文件名中包含test 的文件中不包含test 的行
testfile1:helLinux!
testfile1:Linis a free Unix-type operating system.
testfile1:Lin
testfile_1:HELLO LINUX!
testfile_1:LINUX IS A FREE UNIX-TYPE OPTERATING SYSTEM.
testfile_1:THIS IS A LINUX TESTFILE!
testfile_2:HELLO LINUX!
testfile_2:Linux is a free unix-type opterating system.
```

Linux ispell命令

Linux ispell命令用于拼写检查程序。

ispell预设会使用/usr/lib/ispell/english.hash字典文件来检查文本文件。若在检查的文件中找到字典没有的词汇，ispell会建议使用的词汇，或是让你将新的词汇加入个人字典。

语法

```
ispell [-aAbBClmMnNPStVx][-d<字典文件>][-L<行数>][-p<字典文件>][-w<非字母字符>][-W<字符串长度>][
```

参数：

- -a 当其他程序输出送到ispell时，必须使用此参数。
- -A 读取到"&Include File&"字符串时，就去检查字符串后所指定文件的内容。
- -b 产生备份文件，文件名为.bak。
- -B 检查连字错误。
- -C 不检查连字错误。
- -d<字典文件> 指定字典文件。
- -l 从标准输入设备读取字符串，结束后显示拼错的词汇。
- -L<行数> 指定内文显示的行数。
- -m 自动考虑字尾的变化。
- -M 进入ispell后，在画面下方显示指令的按键。
- -n 检查的文件为noff或troff的格式。
- -N 进入ispell后，在画面下方不显示指令的按键。
- -p<字典文件> 指定个人字典文件。
- -P 不考虑字尾变化的情形。
- -S 不排序建议取代的词汇。
- -t 检查的文件为TeX或LaTeX的格式。
- -V 非ANSI标准的字符会以"M-^"的方式来显示。
- -w<非字母字符> 检查时，特别挑出含有指定的字符。
- -W<字符串长度> 不检查指定长度的词汇。
- -x 不要产生备份文件。

实例

检查文件的拼写。例如，检查testfile文件，可使用如下命令：

```
ispell testfile
```

如果文件中出现可疑词汇，则第一个出现的可疑词汇以高亮显示，并在屏幕下方给出词汇的修改意见，以及ispell的操作命令。如下所示：

```
netwrks File: testfile
Linux netwrks are becoming more and more common, but security is often an overlooked
issue. Unfortunately
0: networks
[SP] <number> R)epl A)ccept I)nsert L)ookup U)ncap Q)uit e(X)it or ? for help
```

本例中，检查出netwrks 错误，并提示纠正信息，此时输入"0"，即使用networks 来纠正错误，同时继续显示下一个错误，直到所有的错误显示完毕。

通过以上实例我们可以发现，文件testfile中有拼写错误，对该文件进行修改后需备份文件。此时使用如下命令：

```
ispell-b testfile    #检查拼写错误的同时，备份文件
```

如果文件已经无拼写错误，则不显示任何信息，通过ls命令我们也可以查看到当前文件目录下产生了文件testfile的备份文件testfile.bak。查看结果如下所示：

```
$ ls #以列表的形式查看当前目录下的文件
examples.desktop testfile_1 testfile.bak xx01 模板图片 音乐
testfile testfile1 testfile_2 xx00 公共的视频文档桌面
```

其中，testfile.bak 文件就是刚才命令生成的备份文件，内容与原来的testfile 文件内容是一样的。

Linux jed命令

Linux jed命令用于编辑文本文件。

Jed是以Slang所写成的程序，适合用来编辑程序原始代码。

语法

```
jed [-2n][-batch][-f<函数>][-g<行数>][-i<文件>][-I<文件>][-s<字符串>][文件]
```

参数：

- -2 显示上下两个编辑区。
- -batch 以批处理模式来执行。
- -f<函数> 执行Slang函数。
- -g<行数> 移到缓冲区中指定的行数。
- -i<文件> 将指定的文件载入缓冲区。
- -i<文件> 载入Slang原始代码文件。
- -n 不要载入jed.rc配置文件。
- -s<字符串> 查找并移到指定的字符串。

实例

jed主要用于编辑程序的源码，编辑源码时将以彩色高亮的方式显示程序的语法。例如使用jed编辑一个C语言的源代码文件，可使用如下命令：

```
jed main.c       #用jed编辑器打开main.c 文件
```

输出结果如下：

```
F10 key ==> File Edit Mode Search Buffers Windows System Help #编辑器菜单
/*-*- linux-c-*-*/ #编辑区
#include <linux/mm.h>
#include <linux/sysctl.h>
#include <linux/nsproxy.h>
static struct list_head *
net_ctl_header_lookup(struct ctl_table_root *root, struct nsproxy *namespaces)
{
return &namespaces->net_ns->sysctl_table_headers;
}
static struct ctl_table_root net_sysctl_root = {
.lookup = net_ctl_header_lookup,
};
static int sysctl_net_init(struct net *net)
{
INIT_LIST_HEAD(&net->sysctl_table_headers);
return 0;
}
-----+(Jed 0.99.18U) Emacs: main.c (C) All 6:06pm-----------------------------
#从左到右分别为jed版本编号、当前是模拟emacs编辑器、打开的文件名、现在的时间
loading /usr/share/jed/lib/modeinfo.slc
```

Linux joe命令

Linux joe命令用于编辑文本文件。

Joe是一个功能强大的全屏幕文本编辑程序。操作的复杂度要比Pico高一点，但是功能较为齐全。Joe一次可开启多个文件，每个文件各放在一个编辑区内，并可在文件之间执行剪贴的动作。

语法

```
joe [-asis][-beep][-csmode][-dopadding][-exask][-force][-help][-keepup][-lightoff][-arkin
```

参数：

- 以下为程序参数 *-asis* 字符码超过*127*的字符不做任何处理。 -backpath<目录> 指定备份文件的目录。* -beep 编辑时，若有错误即发出哔声。

- -columns<栏位> 设置栏数。

- -csmode 可执行连续查找模式。
- -dopadding 是程序跟tty间存在缓冲区。
- -exask 在程序中，执行"Ctrl+k+x"时，会先确认是否要保存文件。
- -force 强制在最后一行的结尾处加上换行符号。
- -help 执行程序时一并显示帮助。
- -keepup 在进入程序后，画面上方为状态列。
- -lightoff 选取的区块在执行完区块命令后，就会回复成原来的状态。
- -lines<行数> 设置行数。
- -marking 在选取区块时，反白区块会随着光标移动。
- -mid 当光标移出画面时，即自动卷页，使光标回到中央。
- -nobackups 不建立备份文件。
- -nonotice 程序执行时，不显示版权信息。
- -nosta 程序执行时，不显示状态列。
- -noxon 尝试取消"Ctrl+s"与"Ctrl+q"键的功能。
- -orphan 若同时开启一个以上的文件，则其他文件会置于独立的缓冲区，而不会另外开启编辑区。
- -pg<行数> 按"PageUp"或"PageDown"换页时，所要保留前一页的行数。
- -skiptop<行数> 不使用屏幕上方指定的行数。
- 以下为文件参数
- +<行数> 指定开启文件时，光标所在的行数。

- -autoindent 自动缩排。
- -crlf 在换行时，使用CR-LF字符。
- -indentc<缩排字符> 执行缩排时，实际插入的字符。
- -istep<缩排字符数> 每次执行缩排时，所移动的缩排字符数。
- -keymap<按键配置文件> 使用不同的按键配置文件。
- -linums 在每行前面加上行号。
- -lmargin<栏数> 设置左侧边界。
- -overwrite 设置覆盖模式。
- -rmargin<栏数> 设置右侧边界。
- -tab<栏数> 设置tab的宽度。
- -rdonly 以只读的方式开启文件-wordwrap编辑时若超过右侧边界，则自动换行。

实例

利用joe命令编辑文本文件。例如利用joe编辑C 语言源代码main.c，使用如下命令：

```
joe main.c
```

与jed类似，joe编辑器中C语言的语法也以彩色的方式显示。效果如下：

```
I A main.c (c) Row 1 Col 1 12:28 Ctrl-K H for help
#上排从左至右分别为打开的文件名、光标所在行列数、现在时间、显示操作说明
/*-*- linux-c-*-*/ #编辑区
#include <linux/mm.h>
#include <linux/sysctl.h>
#include <linux/nsproxy.h>
static struct list_head *
net_ctl_header_lookup(struct ctl_table_root *root, struct nsproxy *namespaces)
{
return &namespaces->net_ns->sysctl_table_headers;
}
static struct ctl_table_root net_sysctl_root = {
.lookup = net_ctl_header_lookup,
};
static int sysctl_net_init(struct net *net)
{
INIT_LIST_HEAD(&net->sysctl_table_headers);
return 0;
}
** Joe's Own Editor v3.5 ** (utf-8) ** Copyright . 2006 ** #joe编辑区的版本及版权信息
```

joe编辑器有一些常用的组合键，例如可以通过Ctrl+K+H 寻求联机帮助，首先按Ctrl+K组合键，再输入字母H，即可调出帮助菜单，通过该帮助信息可以方便地获知如何对joe 编辑器进行操作。

Linux join命令

Linux join命令用于将两个文件中，指定栏位内容相同的行连接起来。

找出两个文件中，指定栏位内容相同的行，并加以合并，再输出到标准输出设备。

语法

```
join [-i][-a<1或2>][-e<字符串>][-o<格式>][-t<字符>][-v<1或2>][-1<栏位>][-2<栏位>][--help][--v
```

参数：

- -a<1或2> 除了显示原来的输出内容之外，还显示指令文件中没有相同栏位的行。
- -e<字符串> 若[文件1]与[文件2]中找不到指定的栏位，则在输出中填入选项中的字符串。
- -i或--igore-case 比较栏位内容时，忽略大小写的差异。
- -o<格式> 按照指定的格式来显示结果。
- -t<字符> 使用栏位的分隔字符。
- -v<1或2> 跟-a相同，但是只显示文件中没有相同栏位的行。
- -1<栏位> 连接[文件1]指定的栏位。
- -2<栏位> 连接[文件2]指定的栏位。
- --help 显示帮助。
- --version 显示版本信息。

实例

连接两个文件。

为了清楚地了解join命令，首先通过cat命令显示文件testfile_1和 testfile_2 的内容。

然后以默认的方式比较两个文件，将两个文件中指定字段的内容相同的行连接起来，在终端中输入命令：

```
join testfile_1 testfile_2
```

首先查看testfile_1、testfile_2 中的文件内容：

```
$ cat testfile_1 #testfile_1文件中的内容
Hello 95 #例如，本例中第一列为姓名，第二列为数额
Linux 85
test 30
cmd@hdd-desktop:~$ cat testfile_2 #testfile_2文件中的内容
Hello 2005 #例如，本例中第一列为姓名，第二列为年份
Linux 2009
test 2006
```

然后使用join命令，将两个文件连接，结果如下：

```
$ join testfile_1 testfile_2 #连接testfile_1、testfile_2中的内容
Hello 95 2005 #连接后显示的内容
Linux 85 2009
test 30 2006
```

文件1与文件2的位置对输出到标准输出的结果是有影响的。例如将命令中的两个文件互换，即输入如下命令：

```
join testfile_2 testfile_1
```

最终在标准输出的输出结果将发生变化，如下所示：

```
$ join testfile_2 testfile_1 #改变文件顺序连接两个文件
Hello 2005 95 #连接后显示的内容
Linux 2009 85
test 2006 30
```

Linux look命令

Linux look命令用于查询单词。

look指令用于英文单字的查询。您仅需给予它欲查询的字首字符串，它会显示所有开头字符串符合该条件的单字。

语法

```
look [-adf][-t<字尾字符串>][字首字符串][字典文件]
```

参数说明：

- -a 使用另一个字典文件web2，该文件也位于/usr/dict目录下。
- -d 只对比英文字母和数字，其余一概忽略不予比对。
- -f 忽略字符大小写差别。
- -t<字尾字符串> 设置字尾字符串。

实例

为了查找在testfile文件中以字母L开头的所有的行，可以输入如下命令：

```
look L testfile
```

原文件testfile中的内容如下：

```
$ cat testfile #查看testfile 文件内容
HELLO LINUX!
Linux is a free unix-type opterating system.
This is a linux testfile!
Linux test
```

在testfile文件中使用look命令查找以"L"开头的单词，结果如下：

```
$ look L testfile                              #查找以“L”开头的单词
Linux is a free unix-type opterating system.   #第二行以“L”开头，列出全句
Linux test                                     #第四行以“L”开头，列出全句
```

Linux mtype命令

mtype为mtools工具指令，模拟MS-DOS的type指令，可显示MS-DOS文件的内容。

语法

```
mtype [-st][文件]
```

参数说明：

- -s 去除8位字符码集的第一个位，使它兼容于7位的ASCII。
- -t 将MS-DOS文本文件中的"换行+光标移至行首"字符转换成Linux的换行字符。

实例

打开名为dos.txt 的MS-DOS文件可使用如下命令：

```
mtype dos.txt         #打开MS-DOS 文件
```

显示结果如下：

```
$ mtype dos.txt #打开MS-DOS 文件
Linux networks are becoming more and more common, but security is often an overlooked
issue. Unfortunately, in today's environment all networks are potential hacker targets,
from top-secret military research networks to small home LANs.
Linux Network Securty focuses on securing Linux in a networked environment, where the
security of the entire network needs to be considered rather than just isolated machines.
It uses a mix of theory and practicl techniques to teach administrators how to install an
use security applications, as well as how the applcations work and why they are necessary
```

Linux pico命令

Linux pico命令用于编辑文字文件。

pico是个简单易用、以显示导向为主的文字编辑程序，它伴随着处理电子邮件和新闻组的程序pine而来。

语法

```
pico [-bdefghjkmqtvwxz][-n<间隔秒数>][-o<工作目录>][-r<编辑页宽>][-s<拼字检查器>][+<列数编号>][
```

参数说明：

- -b 开启置换的功能。
- -d 开启删除的功能。
- -e 使用完整的文件名称。
- -f 支持键盘上的F1、F2...等功能键。
- -g 显示光标。
- -h 在线帮助。
- -j 开启切换的功能。
- -k 预设pico在使用剪下命令时，会把光标所在的列的内容全部删除。
- -m 开启鼠标支持的功能，您可用鼠标点选命令列表。
- -n<间隔秒数> 设置多久检查一次新邮件。
- -o<工作目录> 设置工作目录。
- -q 忽略预设值。
- -r<编辑页宽> 设置编辑文件的页宽。
- -s<拼字检查器> 另外指定拼字检查器。
- -t 启动工具模式。
- -v 启动阅读模式，用户只能观看，无法编辑文件的内容。
- -w 关闭自动换行，通过这个参数可以编辑内容很长的列。
- -x 关闭换面下方的命令列表。
- -z 让pico可被Ctrl+z中断，暂存在后台作业里。
- +<列数编号> 执行pico指令进入编辑模式时，从指定的列数开始编辑。

实例

使用pico命令来编辑testfile文件，在终端中输入如下命令：

```
pico testfile
```

输出结果如下：

```
GNU nano 2.0.9 文件: testfile #从左到右分别为编辑器版本号、文件名
#编辑区
Linux networks are becoming more and more common, but security is often an over$
Linux Network Securty focuses on securing Linux in a networked environment, whe$
[ 已读取3 行] #以下为菜单栏
^G 求助^O 写入^R 读档^Y 上页^K 剪切文字^C 在标位置
^X 离开^J 对齐^W 搜寻^V 下页^U 还原剪切^T 拼写检查
```

Linux rgrep命令

Linux rgrep命令用于递归查找文件里符合条件的字符串。

rgrep指令的功能和grep指令类似，可查找内容包含指定的范本样式的文件，如果发现某文件的内容符合所指定的范本样式，预设rgrep指令会把含有范本样式的那一列显示出来。

语法

```
rgrep [-?BcDFhHilnNrv][-R<范本样式>][-W<列长度>][-x<扩展名>][--help][--version][范本样式][文件
```

参说明数：

- -? 显示范本样式与范例的说明。
- -B 忽略二进制的数据。
- -c 计算符合范本样式的列数。
- -D 排错模式，只列出指令搜寻的目录清单，而不会读取文件内容。
- -F 当遇到符号连接时，rgrep预设是忽略不予处理，加上本参数后，rgrep指令就会读取该连接所指向的原始文件的内容。
- -h 特别将符合范本样式的字符串标示出来。
- -H 只列出符合范本样式的字符串，而非显示整列的内容。
- -i 忽略字符大小写的差别。
- -l 列出文件内容符合指定的范本样式的文件名称。
- -n 在显示符合坊本样式的那一列之前，标示出该列的列数编号。
- -N 不要递归处理。
- -r 递归处理，将指定目录下的所有文件及子目录一并处理。
- -R<范本样式> 此参数的效果和指定"-r"参数类似，但只主力符合范本样式文件名称的文件。
- -v 反转查找。
- -W<列长度> 限制符合范本样式的字符串所在列，必须拥有的字符数。
- -x<扩展名> 只处理符合指定扩展名的文件名称的文件。
- --help 在线帮助。
- --version 显示版本信息。

实例

在当前目录下查找句子中包含"Hello"字符串的文件，可使用如下命令：

```
rgrep Hello *
```

其搜索结果如下：

```
$ rgrep Hello *              #在当前目录下查找句子中包含“Hello”字符串的文件
testfile_1:Hello 95          #testfile_1中包含“Hello”字符串的句子
testfile_2:Hello 2005        #testfile_2中包含“Hello”字符串的句子
```

Linux sed命令

Linux sed命令是利用script来处理文本文件。

sed可依照script的指令，来处理、编辑文本文件。

语法

```
sed [-hnV][-e<script>][-f<script文件>][文本文件]
```

参数说明：

- -e<script>或--expression=<script> 以选项中指定的script来处理输入的文本文件。
- -f<script文件>或--file=<script文件> 以选项中指定的script文件来处理输入的文本文件。
- -h或--help 显示帮助。
- -n或--quiet或--silent 仅显示script处理后的结果。
- -V或--version 显示版本信息。

实例

在testfile文件的第四行后添加一行，并将结果输出到标准输出，在命令行提示符下输入如下命令：

```
sed -e 4a\newLine testfile
```

首先查看testfile中的内容如下：

```
$ cat testfile #查看testfile 中的内容
HELLO LINUX!
Linux is a free unix-type opterating system.
This is a linux testfile!
Linux test
```

使用sed命令后，输出结果如下：

```
$ sed -e 4a\newline testfile #使用sed 在第四行后添加新字符串
HELLO LINUX! #testfile文件原有的内容
Linux is a free unix-type opterating system.
This is a linux testfile!
Linux test
newline
```

Linux sort命令

Linux sort命令用于将文本文件内容加以排序。

sort可针对文本文件的内容，以行为单位来排序。

语法

```
sort [-bcdfimMnr][-o<输出文件>][-t<分隔字符>][+<起始栏位>-<结束栏位>][--help][--verison][文件]
```

参数说明：

- -b 忽略每行前面开始出的空格字符。
- -c 检查文件是否已经按照顺序排序。
- -d 排序时，处理英文字母、数字及空格字符外，忽略其他的字符。
- -f 排序时，将小写字母视为大写字母。
- -i 排序时，除了040至176之间的ASCII字符外，忽略其他的字符。
- -m 将几个排序好的文件进行合并。
- -M 将前面3个字母依照月份的缩写进行排序。
- -n 依照数值的大小排序。
- -o<输出文件> 将排序后的结果存入指定的文件。
- -r 以相反的顺序来排序。
- -t<分隔字符> 指定排序时所用的栏位分隔字符。
- +<起始栏位>-<结束栏位> 以指定的栏位来排序，范围由起始栏位到结束栏位的前一栏位。
- --help 显示帮助。
- --version 显示版本信息。

实例

在使用sort命令以默认的式对文件的行进行排序，使用的命令如下：

```
sort testfile
```

sort 命令将以默认的方式将文本文件的第一列以ASCII 码的次序排列，并将结果输出到标准输出。

使用 cat命令显示testfile文件可知其原有的排序如下：

```
$ cat testfile      #testfile文件原有排序
test 30
Hello 95
Linux 85
```

使用sort命令重排后的结果如下：

```
$ sort testfile #重排结果
Hello 95
Linux 85
test 30
```

Linux spell命令

Linux spell命令可建立拼写检查程序。

spell可从标准输入设备读取字符串，结束后显示拼错的词汇。

语法

```
spell
```

实例

检查文件testfile是否有拼写错误，在命令行提示符下输入如下命令：

```
spell testfile
```

如果文件中有单词拼写错误，则输出如下信息：

```
$ spell testfile   #检查testfile 拼写错误
scurity            #以下为有错误的单词
tp
LANs
Securty
practicl
applcations
necesary
```

如果所检查的文件没有单词拼写错误，那么，命令运行后不会给出任何信息。

检查从标准输入读取的字符串。例如在命令行中输入如下命令：

```
spell
```

按回车键后，输入一串字符串，然后按Ctrl+D 组合键退出spell，屏幕上将显示拼写有错误的单词。如下所示：

```
$ spell #检查标准输入的字符串的拼写错误
hell,this is a linx sustem! #拼写错误的字符串
linx #以下为有拼写错误的单词
sustem
```

Linux tr命令

Linux tr 命令用于转换或删除文件中的字符。

tr 指令从标准输入设备读取数据，经过字符串转译后，将结果输出到标准输出设备。

语法

```
tr [-cdst][--help][--version][第一字符集][第二字符集]
tr [OPTION]…SET1[SET2]
```

参数说明：

- -c, --complement：反选设定字符。也就是符合 SET1 的部份不做处理，不符合的剩余部份才进行转换
- -d, --delete：删除指令字符
- -s, --squeeze-repeats：缩减连续重复的字符成指定的单个字符
- -t, --truncate-set1：削减 SET1 指定范围，使之与 SET2 设定长度相等
- --help：显示程序用法信息
- --version：显示程序本身的版本信息

字符集合的范围：

- \NNN 八进制值的字符 NNN (1 to 3 为八进制值的字符)
- \ 反斜杠
- \a Ctrl-G 铃声
- \b Ctrl-H 退格符
- \f Ctrl-L 走行换页
- \n Ctrl-J 新行
- \r Ctrl-M 回车
- \t Ctrl-I tab键
- \v Ctrl-X 水平制表符
- CHAR1-CHAR2 ：字符范围从 CHAR1 到 CHAR2 的指定，范围的指定以 ASCII 码的次序为基础，只能由小到大，不能由大到小。
- [CHAR*] ：这是 SET2 专用的设定，功能是重复指定的字符到与 SET1 相同长度为止
- [CHAR*REPEAT] ：这也是 SET2 专用的设定，功能是重复指定的字符到设定的 REPEAT 次数为止(REPEAT 的数字采 8 进位制计算，以 0 为开始)
- [:alnum:] ：所有字母字符与数字
- [:alpha:] ：所有字母字符
- [:blank:] ：所有水平空格

- [:cntrl:] ：所有控制字符
- [:digit:] ：所有数字
- [:graph:] ：所有可打印的字符(不包含空格符)
- [:lower:] ：所有小写字母
- [:print:] ：所有可打印的字符(包含空格符)
- [:punct:] ：所有标点字符
- [:space:] ：所有水平与垂直空格符
- [:upper:] ：所有大写字母
- [:xdigit:] ：所有 16 进位制的数字
- [=CHAR=] ：所有符合指定的字符(等号里的 CHAR，代表你可自订的字符)

实例

将文件testfile中的小写字母全部转换成大写字母，此时，可使用如下命令：

```
cat testfile |tr a-z A-Z
```

testfile文件中的内容如下：

```
$ cat testfile         #testfile原来的内容
Linux networks are becoming more and more common,
but scurity is often an overlooked
issue. Unfortunately, in today’s environment all networks
are potential hacker targets,
fro0m tp-secret military research networks to small home LANs.
Linux Network Securty focuses on securing Linux in a
networked environment, where the
security of the entire network needs to be considered
rather than just isolated machines.
It uses a mix of theory and practicl techniques to
teach administrators how to install and
use security applications, as well as how the
applcations work and why they are necesary.
```

使用 tr 命令大小写转换后，得到如下输出结果：

```
$ cat testfile | tr a-z A-Z #转换后的输出
LINUX NETWORKS ARE BECOMING MORE AND MORE COMMON, BUT SCURITY IS OFTEN AN OVERLOOKED
ISSUE. UNFORTUNATELY, IN TODAY’S ENVIRONMENT ALL NETWORKS ARE POTENTIAL HACKER TARGETS,
FROM TP-SECRET MILITARY RESEARCH NETWORKS TO SMALL HOME LANS.
LINUX NETWORK SECURTY FOCUSES ON SECURING LINUX IN A NETWORKED ENVIRONMENT, WHERE THE
SECURITY OF THE ENTIRE NETWORK NEEDS TO BE CONSIDERED RATHER THAN JUST ISOLATED MACHINES.
IT USES A MIX OF THEORY AND PRACTICL TECHNIQUES TO TEACH ADMINISTRATORS HOW TO INSTALL AN
USE SECURITY APPLICATIONS, AS WELL AS HOW THE APPLCATIONS WORK AND WHY THEY ARE NECESARY.
```

大小写转换，也可以通过[:lower][:upper]参数来实现。例如使用如下命令：

```
cat testfile |tr [:lower:] [:upper:]
```

输出结果如下：

```
$ cat testfile | tr [:lower:] [:upper:] #转换后的输出
LINUX NETWORKS ARE BECOMING MORE AND MORE COMMON, BUT SCURITY IS OFTEN AN OVERLOOKED
ISSUE. UNFORTUNATELY, IN TODAY’S ENVIRONMENT ALL NETWORKS ARE POTENTIAL HACKER TARGETS,
FROM TP-SECRET MILITARY RESEARCH NETWORKS TO SMALL HOME LANS.
LINUX NETWORK SECURTY FOCUSES ON SECURING LINUX IN A NETWORKED ENVIRONMENT, WHERE THE
SECURITY OF THE ENTIRE NETWORK NEEDS TO BE CONSIDERED RATHER THAN JUST ISOLATED MACHINES.
IT USES A MIX OF THEORY AND PRACTICL TECHNIQUES TO TEACH ADMINISTRATORS HOW TO INSTALL AN
USE SECURITY APPLICATIONS, AS WELL AS HOW THE APPLCATIONS WORK AND WHY THEY ARE NECESARY.
```

Linux expr命令

expr命令是一个手工命令行计数器，用于在UNIX/LINUX下求表达式变量的值，一般用于整数值，也可用于字符串。

语法

```
expr 表达式
```

表达式说明:

- 用空格隔开每个项；
- 用 / (反斜杠) 放在 shell 特定的字符前面；
- 对包含空格和其他特殊字符的字符串要用引号括起来

实例

1、计算字串长度

```
> expr length “this is a test”
 14
```

2、抓取字串

```
> expr substr “this is a test” 3 5
is is
```

3、抓取第一个字符数字串出现的位置

```
> expr index "sarasara"  a
 2
```

4、整数运算

```
> expr 14 % 9
5
> expr 10 + 10
20
> expr 1000 + 900
1900
> expr 30 / 3 / 2
5
> expr 30 /* 3 (使用乘号时，必须用反斜线屏蔽其特定含义。因为shell可能会误解显示星号的意义)
90
> expr 30 * 3
expr: Syntax error
```

Linux uniq命令

Linux uniq命令用于检查及删除文本文件中重复出现的行列。

uniq可检查文本文件中重复出现的行列。

语法

```
uniq [-cdu][-f<栏位>][-s<字符位置>][-w<字符位置>][--help][--version][输入文件][输出文件]
```

参数：

- -c或--count 在每列旁边显示该行重复出现的次数。
- -d或--repeated 仅显示重复出现的行列。
- -f<栏位>或--skip-fields=<栏位> 忽略比较指定的栏位。
- -s<字符位置>或--skip-chars=<字符位置> 忽略比较指定的字符。
- -u或--unique 仅显示出一次的行列。
- -w<字符位置>或--check-chars=<字符位置> 指定要比较的字符。
- --help 显示帮助。
- --version 显示版本信息。
- [输入文件] 指定已排序好的文本文件。
- [输出文件] 指定输出的文件。

实例

文件testfile中第2 行、第5 行、第9 行为相同的行，使用uniq 命令删除重复的行，可使用以下命令：

```
uniq testfile
```

testfile中的原有内容为：

```
$ cat testfile      #原有内容
test 30
test 30
test 30
Hello 95
Hello 95
Hello 95
Hello 95
Linux 85
Linux 85
```

使用uniq 命令删除重复的行后，有如下输出结果：

```
$ uniq testfile     #删除重复行后的内容
test 30
Hello 95
Linux 85
```

检查文件并删除文件中重复出现的行，并在行首显示该行重复出现的次数。使用如下命令：

```
uniq-c testfile
```

结果输出如下：

```
$ uniq-ctestfile      #删除重复行后的内容
3 test 30             #前面的数字的意义为该行共出现了3次
4 Hello 95            #前面的数字的意义为该行共出现了4次
2 Linux 85            #前面的数字的意义为该行共出现了2次
```

Linux wc命令

Linux wc命令用于计算字数。

利用wc指令我们可以计算文件的Byte数、字数、或是列数，若不指定文件名称、或是所给予的文件名为"-"，则wc指令会从标准输入设备读取数据。

语法

```
wc [-clw][--help][--version][文件...]
```

参数：

- -c或--bytes或--chars 只显示Bytes数。
- -l或--lines 只显示列数。
- -w或--words 只显示字数。
- --help 在线帮助。
- --version 显示版本信息。

实例

在默认的情况下，wc将计算指定文件的行数、字数，以及字节数。使用的命令为：

```
wc testfile
```

先查看testfile文件的内容，可以看到：

```
$ cat testfile
Linux networks are becoming more and more common, but scurity is often an overlooked
issue. Unfortunately, in today’s environment all networks are potential hacker targets,
fro0m tp-secret military research networks to small home LANs.
Linux Network Securty focuses on securing Linux in a networked environment, where the
security of the entire network needs to be considered rather than just isolated machines.
It uses a mix of theory and practicl techniques to teach administrators how to install an
use security applications, as well as how the applcations work and why they are necesary.
```

使用 wc统计，结果如下：

```
$ wc testfile           # testfile文件的统计信息
3 92 598 testfile       # testfile文件的行数为3、单词数92、字节数598
```

其中，3 个数字分别表示testfile文件的行数、单词数，以及该文件的字节数。

如果想同时统计多个文件的信息，例如同时统计testfile、testfile_1、testfile_2，可使用如下命令：

```
wc testfile testfile_1 testfile_2   #统计三个文件的信息
```

输出结果如下：

```
$ wc testfile testfile_1 testfile_2  #统计三个文件的信息
3 92 598 testfile                    #第一个文件行数为3、单词数92、字节数598
9 18 78 testfile_1                   #第二个文件的行数为9、单词数18、字节数78
3 6 32 testfile_2                    #第三个文件的行数为3、单词数6、字节数32
15 116 708 总用量                    #三个文件总共的行数为15、单词数116、字节数708
```

Linux命令大全 - 文件传输

lprm	lpr	lpq	lpd
bye	ftp	uuto	uupick
uucp	uucico	tftp	ncftp
ftpshut	ftpwho	ftpcount	

Linux lprm命令

Linux lprm命令用于将一个工作由打印机贮列中移除

尚未完成的打印机工作会被放在打印机贮列之中，这个命令可用来将常未送到打印机的工作取消。由于每一个打印机都有一个独立的贮列，你可以用 -P 这个命令设定想要作用的印列机。如果没有设定的话，会使用系统预设的打印机。

这个命令会检查使用者是否有足够的权限删除指定的档案，一般而言，只有档案的拥有者或是系统管理员才有这个权限。

语法

```
/usr/bin/lprm [P] [file...]
```

实例

将打印机 hpprinter 中的第 1123 号工作移除

```
lprm -Phpprinter 1123
```

将第 1011 号工作由预设印表机中移除

```
lprm 1011
```

Linux lpr命令

lpr(line printer，按行打印)实用程序用来将一个或多个文件放入打印队列等待打印。

lpr 可以用来将料资送给本地或是远端的主机来处理。

语法

```
lpr [ -P printer ]
```

参数：

- -p Printer: 将资料送至指定的打印机 Printer，预设值为 lp。

实例

下面的命令行将在名为mailroom的打印机上打印report文件：

```
$ lpr -P mailroom report
```

使用一条打印命令可打印多个文件，下面的命令行在名为laser1的打印机上打印3个文件：

```
$ lpr -P laser1 05.txt 108.txt 12.txt
```

Linux lpq命令

Linux lpq命令用于查看一个打印队列的状态，该程序可以查看打印机队列状态及其所包含的打印任务。

语法

lpq [l] [P] [user]

参数说明：

- -P 指定一个打印机，否则使用默认打印机或环境变量PRINTER指定的打印机
- -l 打印组成作业的所有文件的信息。。

实例

为系统默认的打印机printer的一个空队列。

```
$ lpq
printer is ready
no entries
```

如果事先并未指定打印机（使用-P选项），系统便会显示默认的打印机。如果向打印机发送打印任务，然后查看打印队列，便会看到如下列表。

```
$ ls *.txt | pr -3 | lp
request id is printer-603 (1 file(s))
[me@linuxbox ~]$ lpq
printer is ready and printing
Rank    Owner   Job     File(s)                         Total Size
active  me      603     (stdin)
```

Linux lpd命令

Linux lpd命令 是一个常驻的打印机管理程序，它会根据 /etc/printcap 的内容来管理本地或远端的打印机。

/etc/printcap 中定义的每一个打印机必须在 /var/lpd 中有一个相对应的目录，目录中以 cf 开头的档案表示一个等待送到适当装置的印表工作。这个档案通常是由 lpr 所产生。

lpr 和 lpd 组成了一个可以离线工作的系统，当你使用 lpr 时，打印机不需要能立即可用，甚至不用存在。

lpd 会自动监视打印机的状况，当打印机上线后，便立即将档案送交处理。这个得所有的应用程序不必等待打印机完成前一工作。

语法

```
lpd [-l] [#port]
```

参数说明：

- -l: 将一些除错讯息显示在标准输出上。

- # **port:** 一般而言，**lpd** 会使用 **getservbyname** 取得适当的 **TCP/IP port**，你可以使用这个参数强迫 **lpd** 使用指定的 **port**。

实例

这个程序通常是由 /etc/rc.d 中的程序在系统启始阶段执行。

Linux bye命令

Linux bye命令用于中断FTP连线并结束程序。

在ftp模式下，输入bye即可中断目前的连线作业，并结束ftp的执行。

语法

```
bye
```

Linux ftp命令

Linux ftp命令设置文件系统相关功能。

FTP是ARPANet的标准文件传输协议，该网络就是现今Internet的前身。

语法

```
ftp [-dignv][主机名称或IP地址]
```

参数：

- -d 详细显示指令执行过程，便于排错或分析程序执行的情形。
- -i 关闭互动模式，不询问任何问题。
- -g 关闭本地主机文件名称支持特殊字符的扩充特性。
- -n 不使用自动登陆。
- -v 显示指令执行过程。

实例

例如使用ftp命令匿名登录ftp.kernel.org服务器，该服务是Linux 内核的官方服务器，可以使用如下命令：

```
ftp ftp.kernel.org #发起链接请求
```

Linux ncftp命令

Linux ncftp命令用于传输文件。

FTP让用户得以下载存放于服务器主机的文件，也能将文件上传到远端主机放置。

NcFTP是文字模式FTP程序的佼佼者，它具备多样特色， 包括显示传输速率，下载进度，自动续传，标住书签，可通过防火墙和代理服务器等。

当不指定用户名时，ncftp 命令会自动尝试使用匿名账户anonymous 去连接远程FTP 服 务器，不需要用户输入账号和密码。

语法

```
ncftp [主机或IP地址]
```

参数说明：

- -u<用户名> 指定登录FTP服务器的用户名
- -p<密码> 设置用户密码
- -P<端口号> 指定FTP端口号，默认为21
- -j<账号> 指定账号
- -h 帮助信息
- -v 版本信息

实例

使用ncftp命令匿名连接FTP服务器。

例如想匿名连接ftp.kernel.org服务器，同时不想输入anonymous等匿名用户名，可直接使用ncftp命令：

```
ncftp ftp.kernel.org
```

得到如下信息：

```
$ ncftp ftp.kernel.org #匿名连接ftp.kernel.org服务器
NcFTP 3.2.1 (Jul 29, 2007) by Mike Gleason (http://www.NcFTP.com/contact/).
#ncftp版权、版本等信息
Copyright (c) 1992-2005 by Mike Gleason.
All rights reserved.
Connecting to 149.20.20.133... #连接服务器
Welcome to ftp.kernel.org.
Logging in... #匿名登录
Welcome to the #欢迎信息
LINUX KERNEL ARCHIVES
ftp.kernel.org
"Much more than just kernels"
IF YOU'RE ACCESSING THIS SITE VIA A WEB BROWSER
PLEASE USE THE HTTP URL BELOW INSTEAD!
----> If you are looking for mirror sites, please go <----
----> to mirrors.kernel.org instead <----
This site is provided as a public service by the Linux Kernel
Organization, a California nonprofit corporation. Bandwidth is
provided by The Internet Software Consortium, Inc. Our servers are
located in San Francisco and Palo Alto, California; Corvallis, Oregon;
Amsterdam, Netherlands and Ume., Sweden; use in violation of any
applicable laws strictly prohibited.
Due to U.S. Exports Regulations, all cryptographic software on this
site is subject to the following legal notice:
This site includes publicly available encryption source code
which, together with object code resulting from the compiling of
publicly available source code, may be exported from the United
States under License Exception "TSU" pursuant to 15 C.F.R. Section
740.13(e).
This legal notice applies to cryptographic software only. Please see
the Bureau of Industry and Security (http://www.bis.doc.gov/) for more
information about current U.S. regulations.
Neither the Linux Kernel Organization, nor its sponsors make any
guarantees, explicit or implicit, about the contents of this site.
Use at your own risk.
This site is accessible via the following mechanisms:
FTP ftp://ftp.kernel.org/pub/
HTTP http://www.kernel.org/pub/
RSYNC rsync://rsync.kernel.org/pub/
NFS and SMB/CIFS are no longer available.
For comments on this site, please contact <ftpadmin@kernel.org>.
Please do not use this address for questions that are not related to
the operation of this site. Please see our homepage at
http://www.kernel.org/ for links to Linux documentation resources.
Login successful.
Logged in to ftp.kernel.org.
ncftp / >
```

提示：ncftp的命令提示符为"ncftp / >"，而不是ftp中的"ftp / >"。

使用ncftp命令操作、下载文件。

ncftp的命令基本上与ftp相同，例如可以使用"cd"命令切换在FTP服务器中的当前目录，使用"ls"命令列出当前目录内容，使用"get"命令下载"/pub"目录下的README文件、使用"quit"离开ncftp等。操作结果如下：

```
ncftp / > pwd                #查看当前路径
ftp://ftp.kernel.org         #当前路径为根目录
ncftp / > ls                 #查看当前目录列表
bin/ for_mirrors_only/ pub/
dev/ lib/ usr@
etc/ lost+found/ welcome.msg@
ncftp / > cd pub             #切换目录到pub 子目录
Directory successfully changed.
ncftp /pub > ls              #查看pub 的目录列表
dist/ media/ scm/
index.html RCS/ site/
linux/ README software/
lost+found/ README_ABOUT_BZ2_FILES tools/
ncftp /pub > get README      #下载README 文件
README: 1.87 KB 10.39 KB/s
ncftp /pub > quit            #离开ncftp
```

与ftp不同的是，ncftp此时会提示用户是否将FTP服务器保存为书签，以便于下次登录，用户可以进行自定义书签名等操作，如下所示：

```
You have not saved a bookmark for this site.  #离开提示信息
Would you like to save a bookmark to:
ftp://ftp.kernel.org/pub/
Save? (yes/no) yes                            #确认是否保存
Enter a name for this bookmark, or hit enter for "kernel": kernel #输入书签名
Bookmark "kernel" saved.
```

Linux tftp命令

Linux tftp命令用于传输文件。

FTP让用户得以下载存放于远端主机的文件，也能将文件上传到远端主机放置。tftp是简单的文字模式ftp程序，它所使用的指令和FTP类似。

语法

```
tftp [主机名称或IP地址]
```

操作说明：

- connect：连接到远程tftp服务器
- mode：文件传输模式
- put：上传文件
- get：下载文件
- quit：退出
- verbose：显示详细的处理信息
- tarce：显示包路径
- status：显示当前状态信息
- binary：二进制传输模式
- ascii：ascii 传送模式
- rexmt：设置包传输的超时时间
- timeout：设置重传的超时时间
- help：帮助信息
- ? ：帮助信息

实例

连接远程服务器"218.28.188.288"，然后使用put 命令下载其中根目录下的文件"README"，可使用命令如下：

```
tftp 218.28.188.288 #连接远程服务器
```

连接服务器之后可进行相应的操作，具体如下：

```
$ tftp 218.28.188.228                        #连接远程服务器
tftp> ?                                      #使用?，参考帮助
Commands may be abbreviated. Commands are: #帮助命令列表
connect connect to remote tftp
mode set file transfer mode
put send file
get receive file
quit exit tftp
verbose toggle verbose mode
trace toggle packet tracing
status show current status
binary set mode to octet
ascii set mode to netascii
rexmt set per-packet retransmission timeout
timeout set total retransmission timeout
? print help information
tftp>get README                               #远程下载README文件
getting from 218.28.188.288 to /home/cmd
Recived 168236 bytes in 1.5 seconds[112157 bit/s]
tftp>quit                                     #离开tftp
```

Linux uuto命令

Linux uuto命令将文件传送到远端的UUCP主机。

uuto为script文件，它实际上会执行uucp，用来将文件传送到远端UUCP主机，并在完成工作后，以邮件通知远端主机上的用户。

语法

```
uuto [文件][目的]
```

参数：

相关参数请参考 uucp指令。

实例

将文件传送到远程UUCP主机localhost的tmp 目录，在命令提示符中直接输入如下命令：

```
uuto./testfile localhost/tmp #将文件传送到远程UUCP 主机localhost的tmp目录
```

该命令通常没有输出。

Linux uupick命令

Linux uupick命令处理传送进来的文件。

当其他主机通过UUCP将文件传送进来时，可利用uupick指令取出这些文件。

语法

```
uupick [-v][-I<配置文件>][-s<主机>][-x<层级>][--help]
```

参数：

- -I<配置文件>或--config<配置文件> 指定配置文件。
- -s<主机>或--system<主机> 处理由指定主机传送过来的文件。
- -v或--version 显示版本信息。
- --help 显示帮助。

实例

处理由主机localhost传送过来的文件。在命令行直接输入如下命令：

```
uupick-s localhost
```

该命令通常没有输出。

Linux uucp命令

Linux uucp命令用于在Unix系统之间传送文件。

UUCP为Unix系统之间，通过序列线来连线的协议。uucp使用UUCP协议，主要的功能为传送文件。

语法

```
uucp [-cCdfjmrRtvW][-g<等级>][-I<配置文件>][-n<用户>][-x<类型>][--help][...来源][目的]
```

参数说明：

- -c或--nocopy 不用将文件复制到缓冲区。
- -C或--copy 将文件复制到缓冲区。
- -d或--directiories 在传送文件时，自动在[目的]建立必要的目录。
- -f或--nodirectiories 在传送文件时，若需要在[目的]建立目录，则放弃执行该作业。
- -g<等级>或--grade<等级> 指定文件传送作业的优先顺序。
- -I<配置文件>或--config<配置文件> 指定uucp配置文件。
- -j或--jobid 显示作业编号。
- -m或--mail 作业结束后，以电子邮件报告作业是否顺利完成。
- -n<用户>或--notify<用户> 作业结束后，以电子邮件向指定的用户报告作业是否顺利完成。
- -r或--nouucico 不要立即启动uucico服务程序，仅将作业送到队列中，待稍后再执行。
- -R或--recursive 若[来源]为目录，则将整个目录包含子目录复制到[目的]。
- -t或--uuto 将最后一个参数视为"主机名!用户"。
- -v或--version 显示版本信息。
- -W或--noexpand 不要将目前所在的目录加入路径。
- -x<类型>或--debug<类型>启动指定的排错模式。
- --help 显示帮助。
- [源...] 指定源文件或路径。
- [目的] 指定目标文件或路径。

实例

将temp/目录下所有文件传送到远程主机localhost的uucp公共目录下的Public/目录下。在命令行中输入如下命令：

```
uucp-d-R temp localhost ~/Public/
```

该命令通常没有输出

Linux uucico命令

Linux uucico命令UUCP文件传输服务程序。

uucico是用来处理uucp或uux送到队列的文件传输工具。uucico有两种工作模式：主动模式和附属模式。当在主动模式下时，uucico会调用远端主机；在附属模式下时，uucico则接受远端主机的调用。

语法

```
uucico [-cCDefqvwz][-i<类型>][-I<文件>][-p<连接端口号码>][-][-rl][-s<主机>][-S<主机>][-u<用户>
```

参数说明

- -c或--quiet 当不执行任何工作时，不要更改记录文件的内容及更新目前的状态。
- -C或--ifwork 当有工作要执行时，才调用-s或-S参数所指定主机。
- -D或--nodetach 不要与控制终端机离线。
- -e或--loop 在附属模式下执行，并且出现要求登入的提示画面。
- -f或--force 当执行错误时，不等待任何时间即重新调用主机。
- -i<类型>或--stdin<类型> 当使用到标准输入设备时，指定连接端口的类型。
- -I<文件>--config<文件> 指定使用的配置文件。
- -l或--prompt 出现要求登入的提示画面。
- -p<连接端口号码>或-port<连接端口号码> 指定连接端口号码。
- -q或--quiet 不要启动uuxqt服务程序。
- -r0或--slave 以附属模式启动。
- -s<主机>或--system<主机> 调用指定的主机。
- -u<用户>或--login<用户> 指定登入的用户帐号，而不允许输入任意的登入帐号。
- -v或--version 显示版本信息，并且结束程序。
- -w或--wait 在主动模式下，当执行调用动作时，则出现要求登入的提示画面。
- -x<类型>或-X<类型>或outgoing-debug<类型> 启动指定的排错模式。
- -z或--try-next 当执行不成功时，尝试下一个选择而不结束程序。
- --help 显示帮助，并且结束程序。

实例

使用主动模式启动uucico服务。在命令提示符下直接输入如下命令：

```
uucico-r1
```

提示：该命令一般没有输出。

Linux ftpshut命令

Linux ftpshut命令在指定的时间关闭FTP服务器。

本指令提供系统管理者在设置的时间关闭FTP服务器，且能在关闭之前发出警告信息通知用户。关闭时间若设置后为"none"，则会马上关闭服务器。如果采用"+30"的方式来设置表示服务器在30分钟之后关闭。依次类推，假设使用"1130"的格式则代表服务器会在每日的11时30分关闭，时间格式为24小时制。FTP服务器关闭后，在/etc目录下会产生一个名称为shutmsg的文件，把它删除后即可再度启动FTP服务器的功能。

语法

```
ftpshut [-d<分钟>][-l<分钟>][关闭时间]["警告信息"]
```

参数：

- -d<分钟> 切断所有FTP连线时间。
- -l<分钟> 停止接受FTP登入的时间。

实例

在晚上11:00 关闭FTP服务器，并在关闭前5 分钟拒绝新的FTP登录，前3 分钟关闭所有ftp的链接，且给出警告信息，可使用如下命令：

```
ftpshut-d 3 -1 5 1100 "Server will be shutdown at 23:00:00"
```

Linux ftpwho命令

Linux ftpwho命令用于显示目前所有以FTP登入的用户信息。

执行这项指令可得知目前用FTP登入系统的用户有那些人，以及他们正在进行的操作。

语法

```
ftpwho
```

参数说明：

- -v 显示版本信息

实例

查询当前有哪些用户正在登录FTP服务器，可直接使用如下命令：

```
ftpwho
```

该命令有如下输出结果：

```
$ ftpwho              #查询当前正在登录FTP 服务器的用户
standalone FTP daemon[2085]:
3547 wyw [1m20s] 1m25s(idle)
Service class - 1 user #当前有一个用户登录FTP服务器
```

Linux ftpcount命令

Linux ftpcount命令用于显示目前以FTP登入的用户人数。

执行这项指令可得知目前用FTP登入系统的人数以及FTP登入人数的上限。

语法

```
ftpcount
```

参数说明：

- -f<设定文件>：指定设定文件的路径。
- -h，--help ：显示帮助信息。

实例

ftpcount 可以直接查询FTP服务器上用户的人数，可直接使用如下命令：

```
ftpcount              #查询当前FTP用户的人数
```

该命令有如下输出结果：

```
$ ftpcount              #查询当前FTP用户的人数
Master proftpd process 2085:
Service class - 6 user #当前共6个用户登录到服务器
```

Linux命令大全 - 磁盘管理

cd	df	dirs	du
edquota	eject	mcd	mdeltree
mdu	mkdir	mlabel	mmd
mrd	mzip	pwd	quota
mount	mmount	rmdir	rmt
stat	tree	umount	ls
quotacheck	quotaoff	lndir	repquota
quotaon			

Linux cd命令

Linux cd命令用于切换当前工作目录至 dirName(目录参数)。

其中 dirName 表示法可为绝对路径或相对路径。若目录名称省略，则变换至使用者的 home 目录 (也就是刚 login 时所在的目录)。

另外，"~" 也表示为 home 目录 的意思，"." 则是表示目前所在的目录，".." 则表示目前目录位置的上一层目录。

语法

```
cd [dirName]
```

- dirName：要切换的目标目录。

实例

跳到 /usr/bin/ :

```
cd /usr/bin
```

跳到自己的 home 目录 :

```
cd ~
```

跳到目前目录的上上两层 :

```
cd ../..
```

Linux df命令

Linux df命令用于显示目前在Linux系统上的文件系统的磁盘使用情况统计。

语法

```
df [选项]... [FILE]...
```

- 文件-a, --all 包含所有的具有 0 Blocks 的文件系统
- 文件--block-size={SIZE} 使用 {SIZE} 大小的 Blocks
- 文件-h, --human-readable 使用人类可读的格式(预设值是不加这个选项的...)
- 文件-H, --si 很像 -h, 但是用 1000 为单位而不是用 1024
- 文件-i, --inodes 列出 inode 资讯，不列出已使用 block
- 文件-k, --kilobytes 就像是 --block-size=1024
- 文件-l, --local 限制列出的文件结构
- 文件-m, --megabytes 就像 --block-size=1048576
- 文件--no-sync 取得资讯前不 sync (预设值)
- 文件-P, --portability 使用 POSIX 输出格式
- 文件--sync 在取得资讯前 sync
- 文件-t, --type=TYPE 限制列出文件系统的 TYPE
- 文件-T, --print-type 显示文件系统的形式
- 文件-x, --exclude-type=TYPE 限制列出文件系统不要显示 TYPE
- 文件-v (忽略)
- 文件--help 显示这个帮手并且离开
- 文件--version 输出版本资讯并且离开

实例

显示文件系统的磁盘使用情况统计：

```
# df
Filesystem     1K-blocks    Used     Available Use% Mounted on
/dev/sda6       29640780 4320704     23814388  16%     /
udev             1536756       4     1536752    1%     /dev
tmpfs             617620     888     616732     1%     /run
none                5120       0     5120       0%     /run/lock
none             1544044     156     1543888    1%     /run/shm
```

第一列指定文件系统的名称，第二列指定一个特定的文件系统1K-块1K是1024字节为单位的总内存。用和可用列正在使用中，分别指定的内存量。

使用列指定使用的内存的百分比，而最后一栏"安装在"指定的文件系统的挂载点。

df也可以显示磁盘使用的文件系统信息：

```
# df test
Filesystem     1K-blocks    Used      Available Use% Mounted on
/dev/sda6       29640780    4320600   23814492  16%       /
```

用一个-i选项的df命令的输出显示inode信息而非块使用量。

```
df -i
Filesystem      Inodes    IUsed    IFree     IUse% Mounted on
/dev/sda6      1884160    261964   1622196   14%         /
udev           212748     560      212188    1%          /dev
tmpfs          216392     477      215915    1%          /run
none           216392     3        216389    1%          /run/lock
none           216392     8        216384    1%          /run/shm
```

显示所有的信息:

```
# df --total
Filesystem     1K-blocks    Used    Available Use% Mounted on
/dev/sda6       29640780 4320720    23814372  16%     /
udev             1536756       4    1536752   1%      /dev
tmpfs             617620     892    616728    1%      /run
none                5120       0    5120      0%      /run/lock
none             1544044     156    1543888   1%      /run/shm
total           33344320 4321772    27516860  14%
```

我们看到输出的末尾，包含一个额外的行，显示总的每一列。

-h选项，通过它可以产生可读的格式df命令的输出：

```
# df -h
Filesystem      Size  Used   Avail Use% Mounted on
/dev/sda6       29G   4.2G   23G   16%     /
udev            1.5G  4.0K   1.5G   1%     /dev
tmpfs           604M  892K   603M   1%     /run
none            5.0M     0   5.0M   0%     /run/lock
none            1.5G  156K   1.5G   1%     /run/shm
```

我们可以看到输出显示的数字形式的'G'（千兆字节），"M"（兆字节）和"K"（千字节）。

这使输出容易阅读和理解，从而使显示可读的。请注意，第二列的名称也发生了变化，为了使显示可读的"大小"。

Linux dirs命令

Linux dirs命令用于显示目录记录。

显示目录堆叠中的记录。

语法

```
dirs [+/-n -l]
```

参数：

- +n 显示从左边算起第n笔的目录。
- -n 显示从右边算起第n笔的目录。
- -l 显示目录完整的记录。

实例

列出"/home/cc/Ruijie"里所有内容的详细信息。可用如下命令。

```
dir -l /home/cc/Ruijie
```

下面是显示的内容：

```
$ dir -l /home/cc/Ruijie
总计2168
-rwxr-xr-x 1 cc cc  112876 2008-06-26 libpcap.so.0.6.2 -rwxr-xr-x 1 cc cc  737192 2008-06
-rwxr-xr-x 1 cc cc 1350772 2005-08-31 xrgsu
```

Linux du命令

Linux du命令用于显示目录或文件的大小。

du会显示指定的目录或文件所占用的磁盘空间。

语法

```
du [-abcDhHklmsSx][-L <符号连接>][-X <文件>][--block-size][--exclude=<目录或文件>][--max-dept
```

参数说明：

- -a或-all 显示目录中个别文件的大小。
- -b或-bytes 显示目录或文件大小时，以byte为单位。
- -c或--total 除了显示个别目录或文件的大小外，同时也显示所有目录或文件的总和。
- -D或--dereference-args 显示指定符号连接的源文件大小。
- -h或--human-readable 以K，M，G为单位，提高信息的可读性。
- -H或--si 与-h参数相同，但是K，M，G是以1000为换算单位。
- -k或--kilobytes 以1024 bytes为单位。
- -l或--count-links 重复计算硬件连接的文件。
- -L<符号连接>或--dereference<符号连接> 显示选项中所指定符号连接的源文件大小。
- -m或--megabytes 以1MB为单位。
- -s或--summarize 仅显示总计。
- -S或--separate-dirs 显示个别目录的大小时，并不含其子目录的大小。
- -x或--one-file-xystem 以一开始处理时的文件系统为准，若遇上其它不同的文件系统目录则略过。
- -X<文件>或--exclude-from=<文件> 在<文件>指定目录或文件。
- --exclude=<目录或文件> 略过指定的目录或文件。
- --max-depth=<目录层数> 超过指定层数的目录后，予以忽略。
- --help 显示帮助。
- --version 显示版本信息。

实例

显示目录或者文件所占空间:

```
# du
608     ./test6
308     ./test4
4       ./scf/lib
4       ./scf/service/deploy/product
4       ./scf/service/deploy/info
12      ./scf/service/deploy
16      ./scf/service
4       ./scf/doc
4       ./scf/bin
32      ./scf
8       ./test3
1288    .
```

只显示当前目录下面的子目录的目录大小和当前目录的总的大小，最下面的1288为当前目录的总大小

显示指定文件所占空间

```
# du log2012.log
300     log2012.log
```

方便阅读的格式显示test目录所占空间情况：

```
# du -h test
608K    test/test6
308K    test/test4
4.0K    test/scf/lib
4.0K    test/scf/service/deploy/product
4.0K    test/scf/service/deploy/info
12K     test/scf/service/deploy
16K     test/scf/service
4.0K    test/scf/doc
4.0K    test/scf/bin
32K     test/scf
8.0K    test/test3
1.3M    test
```

Linux edquota命令

Linux edquota命令用于编辑用户或群组的磁盘配额。

edquota预设会使用vi来编辑使用者或群组的磁盘配额设置。

语法

```
edquota [-p <源用户名称>][-ug][用户或群组名称...]
```

或

```
edquota [-ug] -t
```

参数：

- -u 设置用户的磁盘配额，这是预设的参数。
- -g 设置群组的磁盘配额。
- -p<源用户名称> 将源用户的磁盘配额设置套用至其他用户或群组。
- -t 设置宽限期限。

Linux mlabel命令

Linux mlabel命令用于设定磁盘的标签 (Label)。

如果磁盘上设定过标签，mlabel 会将他显示给使用者。如果没有指定新标签并且没有指定 c 或 s 选项，mlabel 会提示使用者输入新的标签。如果直接按下 Enter， 就会将原本的标签删除。

语法

```
mlabel [-vcs] drive:[new_label]
```

参数说明：

- -v 更多的讯息。
- -c 清除原有的标签，不出现提示讯息。
- -s 显示目前的标签，不出现提示讯息。

实例

将 A 盘的标签更改为 newlabel。

```
mlabel a:newlabel
```

Linux mkdir命令

Linux mkdir命令用于建立名称为 dirName 之子目录。

语法

```
mkdir [-p] dirName
```

参数说明：

- -p 确保目录名称存在，不存在的就建一个。

实例

在工作目录下，建立一个名为 AAA 的子目录：

```
mkdir AAA
```

在工作目录下的 BBB 目录中，建立一个名为 Test 的子目录。 若 BBB 目录原本不存在，则建立一个。（注：本例若不加 -p，且原本 BBB目录不存在，则产生错误。）

```
mkdir -p BBB/Test
```

Linux mdu命令

Linux mdu命令用于显示MS-DOS目录所占用的磁盘空间。

mdu为mstools工具指令，可显示MS-DOS文件系统中目录所占用的磁盘空间。

语法

```
mdu [-as][目录]
```

参数说明：

- -a 显示每个文件及整个目录所占用的空间。
- -s 仅显示整个目录所占用的空间。

Linux mdeltree命令

Linux mdeltree命令可用来删除 MSDOS 格式档案及目录。

mdeltree 会将所指定的目录与目录之下的所有档案与目录都删除掉。如果所指定的档案或目录不存在，则会传回错误讯息。

语法

```
mdeltree [-v] msdosdirectory [msdosdirectories...]
```

参数说明：

- -v 显示更多的信息。

实例

将 A 磁盘根目录中的 msdosdir 目录以下的档案与目录都删除掉。

```
mcopy a:msdosdir
```

Linux mcd命令

Linux mcd为mtools工具指令，可在MS-DOS文件系统中切换工作目录。若不加任何参数，则显示目前所在的磁盘与工作目录。

语法

```
mcd [msdosdirectory]
```

实例

变更目前工作目录到 a: emp 中。

```
mcd a: emp
```

传回目前工作目录。

```
mcd
```

Linux eject命令

Linux eject命令用于退出抽取式设备。

若设备已挂入，则eject会先将该设备卸除再退出。

语法

```
eject [-dfhnqrstv][-a <开关>][-c <光驱编号>][设备]
```

参数说明：

- [设备] 设备可以是驱动程序名称，也可以是挂入点。
- -a<开关>或--auto<开关> 控制设备的自动退出功能。
- -c<光驱编号>或--changerslut<光驱编号> 选择光驱柜中的光驱。
- -d或--default 显示预设的设备，而不是实际执行动作。
- -f或--floppy 退出抽取式磁盘。
- -h或--help 显示帮助。
- -n或--noop 显示指定的设备。
- -q或--tape 退出磁带。
- -r或--cdrom 退出光盘。
- -s或--scsi 以SCSI指令来退出设备。
- -t或--trayclose 关闭光盘的托盘。
- -v或--verbose 执行时，显示详细的说明。

实例

```
# eject //不加参数默认弹出
# eject -r /dev/cdrom //指定设备
```

Linux mount命令

Linux mount命令是经常会使用到的命令，它用于挂载Linux系统外的文件。

语法

```
mount [-hV]
mount -a [-fFnrsvw] [-t vfstype]
mount [-fnrsvw] [-o options [,...]] device | dir
mount [-fnrsvw] [-t vfstype] [-o options] device dir
```

参数说明：

- -V：显示程序版本
- -h：显示辅助讯息
- -v：显示较讯息，通常和 -f 用来除错。
- -a：将 /etc/fstab 中定义的所有档案系统挂上。
- -F：这个命令通常和 -a 一起使用，它会为每一个 mount 的动作产生一个行程负责执行。在系统需要挂上大量 NFS 档案系统时可以加快挂上的动作。
- -f：通常用在除错的用途。它会使 mount 并不执行实际挂上的动作，而是模拟整个挂上的过程。通常会和 -v 一起使用。
- -n：一般而言，mount 在挂上后会在 /etc/mtab 中写入一笔资料。但在系统中没有可写入档案系统存在的情况下可以用这个选项取消这个动作。
- -s-r：等于 -o ro
- -w：等于 -o rw
- -L：将含有特定标签的硬盘分割挂上。
- -U：将档案分割序号为 的档案系统挂下。-L 和 -U 必须在/proc/partition 这种档案存在时才有意义。
- -t：指定档案系统的型态，通常不必指定。mount 会自动选择正确的型态。
- -o async：打开非同步模式，所有的档案读写动作都会用非同步模式执行。
- -o sync：在同步模式下执行。
- -o atime、-o noatime：当 atime 打开时，系统会在每次读取档案时更新档案的『上一次调用时间』。当我们使用 flash 档案系统时可能会选项把这个选项关闭以减少写入的次数。
- -o auto、-o noauto：打开/关闭自动挂上模式。
- -o defaults:使用预设的选项 rw, suid, dev, exec, auto, nouser, and async.
- -o dev、-o nodev-o exec、-o noexec允许执行档被执行。
- -o suid、-o nosuid：
- 允许执行档在 root 权限下执行。
- -o user、-o nouser：使用者可以执行 mount/umount 的动作。

- -o remount：将一个已经挂下的档案系统重新用不同的方式挂上。例如原先是唯读的系统，现在用可读写的模式重新挂上。
- -o ro：用唯读模式挂上。
- -o rw：用可读写模式挂上。
- -o loop=：使用 loop 模式用来将一个档案当成硬盘分割挂上系统。

实例

将 /dev/hda1 挂在 /mnt 之下。

```
#mount /dev/hda1 /mnt
```

将 /dev/hda1 用唯读模式挂在 /mnt 之下。

```
#mount -o ro /dev/hda1 /mnt
```

将 /tmp/image.iso 这个光碟的 image 档使用 loop 模式挂在 /mnt/cdrom之下。用这种方法可以将一般网络上可以找到的 Linux 光 碟 ISO 档在不烧录成光碟的情况下检视其内容。

```
#mount -o loop /tmp/image.iso /mnt/cdrom
```

Linux mmd命令

Linux mmd命令用于在MS-DOS文件系统中建立目录。

mmd为mtools工具指令，模拟MS-DOS的md指令，可在MS-DOS的文件系统中建立目录。

语法

```
mmd [目录...]
```

Linux mrd命令

Linux mrd命令用于删除MS-DOS文件系统中的目录。

mrd为mtools工具指令，模拟MS-DOS的rd指令，可删除MS-DOS的目录。

语法

```
mrd [目录...]
```

Linux mzip命令

Linux mzip命令是Zip/Jaz磁盘驱动器控制指令。

mzip为mtools工具指令，可设置Zip或Jaz磁盘驱动区的保护模式以及执行退出磁盘的动作。

语法

```
mzip [-efpqruwx]
```

参数：

- -e 退出磁盘。
- -f 与-e参数一并使用，不管是否已经挂入磁盘中的文件系统，一律强制退出磁盘。
- -p 设置磁盘的写入密码。
- -q 显示目前的状态。
- -r 将磁盘设为防写状态。
- -u 退出磁盘以前，暂时解除磁盘的保护状态。
- -w 将磁盘设为可写入状态。
- -x 设置磁盘的密码。

Linux pwd命令

Linux pwd命令用于显示工作目录。

执行pwd指令可立刻得知您目前所在的工作目录的绝对路径名称。

语法

```
pwd [--help][--version]
```

参数说明:

- --help 在线帮助。
- --version 显示版本信息。

实例

查看当前所在目录：

```
# pwd
/root/test            #输出结果
```

Linux quota命令

Linux quota命令用于显示磁盘已使用的空间与限制。

执行quota指令，可查询磁盘空间的限制，并得知已使用多少空间。

语法

```
quota [-quvV][用户名称...] 或 quota [-gqvV][群组名称...]
```

参数说明：

- -g 列出群组的磁盘空间限制。
- -q 简明列表，只列出超过限制的部分。
- -u 列出用户的磁盘空间限制。
- -v 显示该用户或群组，在所有挂入系统的存储设备的空间限制。
- -V 显示版本信息。

实例

```
# quota -guvs    <==显示目前执行者（就是 root ）的 quota 值
# quota -uvs test <==显示 test 这个使用者的 quota 值
```

Linux mmount命令

Linux mmount命令用于挂入MS-DOS文件系统。

mmount为mtools工具指令，可根据[mount参数]中的设置，将磁盘内容挂入到Linux目录中。

语法

```
mmount [驱动器代号][mount参数]
```

参数：

- [mount参数]的用法请参考 mount指令。

Linux rmdir命令

Linux rmdir命令删除空的目录。

语法

```
rmdir [-p] dirName
```

参数：

- -p 是当子目录被删除后使它也成为空目录的话，则顺便一并删除。

实例

将工作目录下，名为 AAA 的子目录删除 :

```
rmdir AAA
```

在工作目录下的 BBB 目录中，删除名为 Test 的子目录。若 Test 删除后，BBB 目录成为空目录，则 BBB 亦予删除。

```
rmdir -p BBB/Test
```

Linux rmt命令

Linux rmt命令通过进程间通信远程控制磁带机。

通过rmt指令，用户可通过IPC连线，远端操控磁带机的倾倒和还原操作。

语法

```
rmt
```

Linux stat命令

Linux stat命令用于显示inode内容。

stat以文字的格式来显示inode的内容。

语法

```
stat [文件或目录]
```

实例

查看 testfile 文件的inode内容内容，可以用以下命令：

stat testfile

执行以上命令输出结果：

```
# stat testfile                #输入命令
  File: `testfile'
  Size: 102             Blocks: 8          IO Block: 4096   regular file
Device: 807h/2055d      Inode: 1265161     Links: 1
Access: (0644/-rw-r--r--)  Uid: (    0/    root)   Gid: (    0/    root)
Access: 2014-08-13 14:07:20.000000000 +0800
Modify: 2014-08-13 14:07:07.000000000 +0800
Change: 2014-08-13 14:07:07.000000000 +0800
```

Linux tree命令

Linux tree命令用于以树状图列出目录的内容。

执行tree指令，它会列出指定目录下的所有文件，包括子目录里的文件。

语法

```
tree [-aACdDfFgilnNpqstux][-I <范本样式>][-P <范本样式>][目录...]
```

参数说明：

- -a 显示所有文件和目录。
- -A 使用ASNI绘图字符显示树状图而非以ASCII字符组合。
- -C 在文件和目录清单加上色彩，便于区分各种类型。
- -d 显示目录名称而非内容。
- -D 列出文件或目录的更改时间。
- -f 在每个文件或目录之前，显示完整的相对路径名称。
- -F 在执行文件，目录，Socket，符号连接，管道名称名称，各自加上"*","/","=","@","|"号。
- -g 列出文件或目录的所属群组名称，没有对应的名称时，则显示群组识别码。
- -i 不以阶梯状列出文件或目录名称。
- -I<范本样式> 不显示符合范本样式的文件或目录名称。
- -l 如遇到性质为符号连接的目录，直接列出该连接所指向的原始目录。
- -n 不在文件和目录清单加上色彩。
- -N 直接列出文件和目录名称，包括控制字符。
- -p 列出权限标示。
- -P<范本样式> 只显示符合范本样式的文件或目录名称。
- -q 用"?"号取代控制字符，列出文件和目录名称。
- -s 列出文件或目录大小。
- -t 用文件和目录的更改时间排序。
- -u 列出文件或目录的拥有者名称，没有对应的名称时，则显示用户识别码。
- -x 将范围局限在现行的文件系统中，若指定目录下的某些子目录，其存放于另一个文件系统上，则将该子目录予以排除在寻找范围外。

实例

以树状图列出当前目录结构。可直接使用如下命令：

```
tree
```

该命令有如下输出结果：

```
# tree                   #以树状图列出当前目录结构
.                        #当前目录结构
|-- README
|-- examples.desktop
|-- file
|-- file.new
|-- index.htm
|-- test
| |-- README
| |-- file
| |-- testfile
| |-- testfile1
| |-- xaa
| |-- xab
| |-- xac
| |-- xad
| |-- xae
| |-- xaf
| |-- xag
| |-- xah
| `-- xai
|-- test.tar.gz
|-- test.zip
|-- testfile
|-- testfile.new
|-- testfile.patch
|-- testfile1
|-- testfile2
|-- testfile3
|-- xaa
|-- xab
|-- xac
|-- xad
|-- xae
|-- xaf
|-- xag
|-- xah
|-- xai
|-- \345\205\254\345\205\261\347\232\204
|-- \345\233\276\347\211\207
| |-- 075b5c2bb1628c1a5343c10a.jpg
| |-- 0c978fe989ac787e799757095719d3c4.jpg
| |-- 2005072619482686643.jpg
| |-- 2006111317154878512.jpg
| |-- 2007102221576687.jpg
| |-- 39.jpg
| |-- 434887ec4340916a78f0559a.jpg
| |-- 498da016ac02fb2bc93d6d08.jpg
| |-- 7b284f5a0f854da2f3bf90b204149a34.jpg
| |-- 9196c030d342a68d5edf0e98.jpg
| |-- a56c5a90de15c8a9a977a4cc.jpg
| |-- c74f62167c9d2b244a90a79e.jpg
| `-- img13.jpg
|-- \346\226\207\346\241\243
|-- \346\241\214\351\235\242
|-- \346\250\241\346\235\277
|-- \350\247\206\351\242\221
`-- \351\237\263\344\271\220
8 directories, 48 files           #统计信息，该目录共8个子目录，48个文件
```

Linux umount命令

Linux umount命令用于卸除文件系统。

umount可卸除目前挂在Linux目录中的文件系统。

语法

```
umount [-ahnrvV][-t <文件系统类型>][文件系统]
```

参数：

- -a 卸除/etc/mtab中记录的所有文件系统。
- -h 显示帮助。
- -n 卸除时不要将信息存入/etc/mtab文件中。
- -r 若无法成功卸除，则尝试以只读的方式重新挂入文件系统。
- -t<文件系统类型> 仅卸除选项中所指定的文件系统。
- -v 执行时显示详细的信息。
- -V 显示版本信息。
- [文件系统] 除了直接指定文件系统外，也可以用设备名称或挂入点来表示文件系统。

实例

下面两条命令分别通过设备名和挂载点卸载文件系统，同时输出详细信息：

```
# umount -v /dev/sda1          通过设备名卸载
/dev/sda1 umounted
# umount -v /mnt/mymount/      通过挂载点卸载
/tmp/diskboot.img umounted
```

如果设备正忙，卸载即告失败。卸载失败的常见原因是，某个打开的shell当前目录为挂载点里的某个目录：

```
# umount -v /mnt/mymount/
umount: /mnt/mymount: device is busy
umount: /mnt/mymount: device is busy
```

Linux ls命令

Linux ls命令用于显示指定工作目录下之内容（列出目前工作目录所含之文件及子目录)。

语法

```
ls [-alrtAFR] [name...]
```

参数 :

- -a 显示所有文件及目录 (ls内定将文件名或目录名称开头为"."的视为隐藏档，不会列出)
- -l 除文件名称外，亦将文件型态、权限、拥有者、文件大小等资讯详细列出
- -r 将文件以相反次序显示(原定依英文字母次序)
- -t 将文件依建立时间之先后次序列出
- -A 同 -a ，但不列出 "." (目前目录) 及 ".." (父目录)
- -F 在列出的文件名称后加一符号；例如可执行档则加 "*", 目录则加 "/"
- -R 若目录下有文件，则以下之文件亦皆依序列出

实例

列出根目录()下的所有目录：

```
# ls /
bin               dev   lib         media  net   root     srv  upload  www
boot              etc   lib64       misc   opt   sbin     sys  usr
home  lost+found  mnt    proc  selinux  tmp  var
```

列出目前工作目录下所有名称是 s 开头的文件，越新的排越后面 :

```
ls -ltr s*
```

将 /bin 目录以下所有目录及文件详细资料列出 :

```
ls -lR /bin
```

列出目前工作目录下所有文件及目录；目录于名称后加 "/", 可执行档于名称后加 "*" :

```
ls -AF
```

Linux quotacheck命令

Linux quotacheck命令用于检查磁盘的使用空间与限制。

执行quotacheck指令，扫描挂入系统的分区，并在各分区的文件系统根目录下产生quota.user和quota.group文件，设置用户和群组的磁盘空间限制。

语法

```
quotacheck [-adgRuv][文件系统...]
```

参数：

- -a 扫描在/etc/fstab文件里，有加入quota设置的分区。
- -d 详细显示指令执行过程，便于排错或了解程序执行的情形。
- -g 扫描磁盘空间时，计算每个群组识别码所占用的目录和文件数目。
- -R 排除根目录所在的分区。
- -u 扫描磁盘空间时，计算每个用户识别码所占用的目录和文件数目。
- -v 显示指令执行过程。

Linux quotaoff命令

Linux quotaoff命令关闭磁盘空间限制。

执行quotaoff指令可关闭用户和群组的磁盘空间限制。

语法

```
quotaoff [-aguv][文件系统...]
```

参数说明：

- -a 关闭在/etc/fstab文件里，有加入quota设置的分区的空间限制。
- -g 关闭群组的磁盘空间限制。
- -u 关闭用户的磁盘空间限制。
- -v 显示指令执行过程。

实例

关闭配额限制:

```
# quotaoff -a
```

Linux lndir命令

Linux lndir命令用于连接目录内容。

执行lndir指令，可一口气把源目录底下的文件和子目录统统建立起相互对应的符号连接。

语法

```
lndir [-ignorelinks][-silent][源目录][目的目录]
```

参数：

- -ignorelinks 直接建立符号连接的符号连接。
- -silent 不显示指令执行过程。

实例

给目录下所有的文件或者子文件目录建立链接：

```
lndir /home/uptech abc
```

Linux repquota命令

Linux repquota命令用于检查磁盘空间限制的状态。

执行repquota指令，可报告磁盘空间限制的状况，清楚得知每位用户或每个群组已使用多少空间。

语法

```
repquota [-aguv][文件系统...]
```

参数说明：

- -a 列出在/etc/fstab文件里，有加入quota设置的分区的使用状况，包括用户和群组。
- -g 列出所有群组的磁盘空间限制。
- -u 列出所有用户的磁盘空间限制。
- -v 显示该用户或群组的所有空间限制。

Linux quotaon命令

Linux quotaon命令用于开启磁盘空间限制。

执行quotaon指令可开启用户和群组的才磅秒年空间限制，各分区的文件系统根目录必须有quota.user和quota.group配置文件。

语法

```
quotaon [-aguv][文件系统...]
```

参数说明：

- -a 开启在/ect/fstab文件里，有加入quota设置的分区的空间限制。
- -g 开启群组的磁盘空间限制。
- -u 开启用户的磁盘空间限制。
- -v 显示指令指令执行过程。

Linux命令大全 - 磁盘维护

badblocks	cfdisk	dd	e2fsck
ext2ed	fsck	fsck.minix	fsconf
fdformat	hdparm	mformat	mkbootdisk
mkdosfs	mke2fs	mkfs.ext2	mkfs.msdos
mkinitrd	mkisofs	mkswap	mpartition
swapon	symlinks	sync	mbadblocks
mkfs.minix	fsck.ext2	fdisk	losetup
mkfs	sfdisk	swapoff	

Linux badblocks命令

Linux badblocks命令用于检查磁盘装置中损坏的区块。

执行指令时须指定所要检查的磁盘装置，及此装置的磁盘区块数。

语法

```
badblocks [-svw][-b <区块大小>][-o <输出文件>][磁盘装置][磁盘区块数][启始区块]
```

参数说明：

- -b<区块大小> 指定磁盘的区块大小，单位为字节。
- -o<输出文件> 将检查的结果写入指定的输出文件。
- -s 在检查时显示进度。
- -v 执行时显示详细的信息。
- -w 在检查时，执行写入测试。
- [磁盘装置] 指定要检查的磁盘装置。
- [磁盘区块数] 指定磁盘装置的区块总数。
- [启始区块] 指定要从哪个区块开始检查。

实例

查看系统当前硬盘信息。

```
# fdisk -l
```

例如，显示信息如下：

```
Disk /dev/sda: 298.9 GB, 298999349248 bytes
255 heads, 63 sectors/track, 36351 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

  Device Boot      Start         End      Blocks   Id  System
/dev/sda1               1         262     2104483+  82  Linux swap / Solaris
/dev/sda2   *         263       32898   262148670   83  Linux
/dev/sda3           32899       36351    27736222+  83  Linux

Disk /dev/sdb: 42.9 GB, 42949672960 bytes
64 heads, 32 sectors/track, 40960 cylinders
Units = cylinders of 2048 * 512 = 1048576 bytes
```

通过命令扫描硬盘。

```
# badblocks -s -v /dev/sdnx
```

其中n表示硬盘设备名，x表示硬盘对应的分区号。例如需要检查"/dev/sda2"，执行命令如下：

```
# badblocks -s -v /dev/sda2

Checking blocks 0 to 30681000
Checking for bad blocks (read-only test): 306809600674112/ 306810000000
30680964
30680965
30680966
30680967
30680968
30680969
30680970
30680971
30680972
30680973
...
done
Pass completed, 37 bad blocks found.其中，“37 bad blocks found”表示硬盘存在37个坏块。
```

Linux cfdisk命令

Linux cfdisk命令用于磁盘分区。

cfdisk是用来磁盘分区的程序，它十分类似DOS的fdisk，具有互动式操作界面而非传统fdisk的问答式界面，您可以轻易地利用方向键来操控分区操作。

语法

```
cfdisk [-avz][-c <柱面数目>-h <磁头数目>-s <盘区数目>][-P <r,s,t>][外围设备代号]
```

参数说明：

- -a 在程序里不用反白代表选取，而以箭头表示。
- -c<柱面数目> 忽略BIOS的数值，直接指定磁盘的柱面数目。
- -h<磁头数目> 忽略BIOS的数值，直接指定磁盘的磁头数目。
- -P<r,s,t> 显示分区表的内容，附加参数"r"会显示整个分区表的详细资料，附加参数"s"会依照磁区的顺序显示相关信息，附加参数"t"则会以磁头，磁区，柱面的方式来显示资料。
- -s<磁区数目> 忽略BIOS的数值，直接指定磁盘的磁区数目。
- -v 显示版本信息。
- -z 不读取现有的分区，直接当作没有分区的新磁盘使用。

实例

进行磁盘分区：

```
# cfsik
```

进行磁盘分区，使用箭头进行操作，而不使用反白表示：

```
# cfsik -a
```

进行磁盘分区，使用箭头进行操作，而不使用反白表示：

```
# cfsik -s 3
```

VM59220:1 Resource interpreted as Image but transferred with MIME type text/html: "http://googleads.g.doubleclick.net/pagead/adview?ai=C9xWpIRctVK75CYm28gXc8Y…bC1FEwdf3NGxgwxaMy7yS-2Ac7otU34AGyoyh--HWiPSLAaAGIQ&sigh=-A-s3VVItog&vis=1". ads?client=ca-pub-5751451760833794

Linux dd命令

Linux dd命令用于读取、转换并输出数据。

dd可从标准输入或文件中读取数据，根据指定的格式来转换数据，再输出到文件、设备或标准输出。

参数说明:

- if=文件名：输入文件名，缺省为标准输入。即指定源文件。
- of=文件名：输出文件名，缺省为标准输出。即指定目的文件。
- ibs=bytes：一次读入bytes个字节，即指定一个块大小为bytes个字节。
  obs=bytes：一次输出bytes个字节，即指定一个块大小为bytes个字节。
  bs=bytes：同时设置读入/输出的块大小为bytes个字节。
- cbs=bytes：一次转换bytes个字节，即指定转换缓冲区大小。
- skip=blocks：从输入文件开头跳过blocks个块后再开始复制。
- seek=blocks：从输出文件开头跳过blocks个块后再开始复制。
- count=blocks：仅拷贝blocks个块，块大小等于ibs指定的字节数。
- conv=<关键字>，关键字可以有以下11种：
  - conversion：用指定的参数转换文件。
  - ascii：转换ebcdic为ascii
  - ebcdic：转换ascii为ebcdic
  - ibm：转换ascii为alternate ebcdic
  - block：把每一行转换为长度为cbs，不足部分用空格填充
  - unblock：使每一行的长度都为cbs，不足部分用空格填充
  - lcase：把大写字符转换为小写字符
  - ucase：把小写字符转换为大写字符
  - swab：交换输入的每对字节
  - noerror：出错时不停止
  - notrunc：不截短输出文件
  - sync：将每个输入块填充到ibs个字节，不足部分用空（NUL）字符补齐。
- --help：显示帮助信息
- --version：显示版本信息

实例

在Linux 下制作启动盘，可使用如下命令：

```
dd if=boot.img of=/dev/fd0 bs=1440k
```

将testfile文件中的所有英文字母转换为大写，然后转成为testfile_1文件，在命令提示符中使用如下命令：

```
dd if=testfile_2 of=testfile_1 conv=ucase
```

其中testfile_2 的内容为：

```
$ cat testfile_2 #testfile_2的内容
HELLO LINUX!
Linux is a free unix-type opterating system.
This is a linux testfile!
Linux test
```

转换完成后，testfile_1 的内容如下：

```
$ dd if=testfile_2 of=testfile_1 conv=ucase #使用dd 命令，大小写转换记录了0+1 的读入
记录了0+1 的写出
95字节（95 B）已复制，0.000131446 秒，723 KB/s
cmd@hdd-desktop:~$ cat testfile_1 #查看转换后的testfile_1文件内容
HELLO LINUX!
LINUX IS A FREE UNIX-TYPE OPTERATING SYSTEM.
THIS IS A LINUX TESTFILE!
LINUX TEST #testfile_2中的所有字符都变成了大写字母
```

由标准输入设备读入字符串，并将字符串转换成大写后，再输出到标准输出设备，使用的命令为：

```
dd conv=ucase
```

输入以上命令后按回车键，输入字符串，再按回车键，按组合键Ctrl+D 退出，出现以下结果：

```
$ dd conv=ucase
Hello Linux! #输入字符串后按回车键
HELLO LINUX! #按组合键Ctrl+D退出，转换成大写结果
记录了0+1 的读入
记录了0+1 的写出
13字节（13 B）已复制，12.1558 秒，0.0 KB/s
```

href http://www.w3cschool.cc/linux/linux-comm-e2fsck.html linux-comm-e2fsck.html:31 'Attr.nodeValue' is deprecated. Please use 'value' instead. adsbygoogle.js:32

Linux e2fsck命令

Linux e2fsck命令用于检查使用 Linux ext2 档案系统的 partition 是否正常工作。

语法

```
e2fsck [-pacnydfvFV] [-b superblock] [-B blocksize] [-l|-L bad_blocks_file] [-C fd] devic
```

参数说明：

- device ： 预备检查的硬盘 partition，例如：/dev/sda1
- -a : 对 partition 做检查，若有问题便自动修复，等同 -p 的功能
- -b : 设定存放 superblock 的位置
- -B : 设定单位 block 的大小
- -c : 检查该partition 是否有坏轨
- -C file : 将检查的结果存到 file 中以便查看
- -d : 列印 e2fsck 的 debug 结果
- -f : 强制检查
- -F : 在开始检查前，将device 的 buffer cache 清空，避免有错误发生
- -l bad_blocks_file : 将有坏轨的block资料加到 bad_blocks_file 里面
- -L bad_blocks_file : 设定坏轨的block资料存到 bad_blocks_file 里面，若无该档则自动产生
- -n : 将档案系统以[唯读]方式开启
- -p : 对 partition 做检查，若有问题便自动修复
- -v : 详细显示模式
- -V : 显示出目前 e2fsck 的版本
- -y : 预先设定所有检查时的问题均回答[是]

实例

检查 /dev/hda5 是否正常，如果有异常便自动修复，并且设定若有问答，均回答[是] :

```
e2fsck -a -y /dev/hda5
```

注意 :

大部份使用 e2fsck 来检查硬盘 partition 的情况时，通常都是情形特殊，因此最好先将该 partition umount，然后再执行 e2fsck 来做检查，若是要非要检查 / 时，则请进入 singal user mode 再执行。

Linux ext2ed命令

Linux ext2ed命令是ext2文件系统编辑程序。

ext2ed可直接处理硬盘分区上的数据，这指令只有Red Hat Linux才提供。

语法

```
ext2ed
```

一般指令：

- setdevice[设备名称] 指定要处理的设备。
- disablewrite 将ext2ed设为只读的状态。
- enablewrite 将ext2ed设为可读写的状态。
- help[指令] 显示个别指令的帮助。
- next 移至下一个单位，单位会依目前所在的模式而异。
- prev 移至前一个单位，单位会依目前所在的模式而异。
- pgup 移至下一页。
- pgdn 移至上一页。
- set 修改目前的数据，参数会依目前所在的模式而异。
- writedata 在执行此指令之后，才会实际修改分区中的数据。
- ext2进入3种模式的指令
- super 进入main superblock,即Superblock模式。
- group<编号> 进入指定的group，即Group模式。
- cd<目录或文件> 在inode模式下，进入指定的目录或文件，即Inode模式。
- Superblock模式
- gocopy<备份编号> 进入指定的superblock备份。
- setactivecopy 将目前所在的superblock，复制到main superblock。
- Group模式
- blockbitmap 显示目前groupo的区块图。
- inode 进入目前group的第一个inode。
- inodebitmap 显示目前group的inode二进制码。
- Inode模式
- dir 进入目录模式。
- file 进入文件模式。

Linux mkbootdisk命令

Linux mkbootdisk命令用于建立目前系统的启动盘。

mkbootdisk可建立目前系统的启动盘。

语法

```
mkbootdisk [--noprompt][--verbose][--version][--device <设备>][--mkinitrdargs <参数>][kern
```

参数：

- --device<设备> 指定设备。
- --mkinitrdargs<参数> 设置mkinitrd的参数。
- --noprompt 不会提示用户插入磁盘。
- --verbose 执行时显示详细的信息。
- --version 显示版本信息。

Linux fsck命令

Linux fsck命令用于检查与修复 Linux 档案系统，可以同时检查一个或多个 Linux 档案系统。

语法

```
fsck [-sACVRP] [-t fstype] [--] [fsck-options] filesys [...]
```

参数 ：

- filesys ： device 名称(eg./dev/sda1)，mount 点 (eg. / 或 /usr)
- -t : 给定档案系统的型式，若在 /etc/fstab 中已有定义或 kernel 本身已支援的则不需加上此参数
- -s : 依序一个一个地执行 fsck 的指令来检查
- -A : 对/etc/fstab 中所有列出来的 partition 做检查
- -C : 显示完整的检查进度
- -d : 列印 e2fsck 的 debug 结果
- -p : 同时有 -A 条件时，同时有多个 fsck 的检查一起执行
- -R : 同时有 -A 条件时，省略 / 不检查
- -V : 详细显示模式
- -a : 如果检查有错则自动修复
- -r : 如果检查有错则由使用者回答是否修复

实例

检查 msdos 档案系统的 /dev/hda5 是否正常，如果有异常便自动修复 :

```
fsck -t msdos -a /dev/hda5
```

注意 此指令可与 /etc/fstab 相互参考操作来加以了解。

Linux fsck.minix命令

Linux fsck.minix命令用于检查文件系统并尝试修复错误。

当minix文件系统发生错误时，可用fsck.minix指令尝试加以参考。

语法

```
fsck.minix [-aflmrsv][外围设备代号]
```

参数：

- -a 自动修复文件系统，不询问任何问题。
- -f 强制对该文件系统进行完整检查，纵然该文件系统在概略检查下没有问题。
- -l 列出所有文件名称。
- -m 使用类似MINIX操作系统的警告信息。
- -r 采用互动模式，在执行修复时询问问题，让用户得以确认并决定处理方式。
- -s 显示该分区第一个磁区的相关信息。
- -v 显示指令执行过程。

Linux fsconf命令

Linux fsconf命令用于设置文件系统相关功能。

fsconf是Red Hat Linux发行版专门用来调整Linux各项设置的程序。

语法

```
fsconf [--check]
```

参数：

- --chedk 检查特定文件的权限。

Linux fdformat命令

Linux fdformat命令用于对指定的软碟机装置进行低阶格式化。

使用这个指令对软碟格式化的时候，最好指定像是下面的装置：

- /dev/fd0d360 磁碟机 A:，磁片为 360KB 磁碟
- /dev/fd0h1440 磁碟机 A:，磁片为 1.4MB 磁碟
- /dev/fd1h1200 磁碟机 B:，磁片为 1.2MB 磁碟

如果使用像是 /dev/fd0 之类的装置，如果里面的磁碟不是标准容量，格式化可能会失败。在这种情况之下，使用者可以用 setfdprm 指令先行指定必要参数。

语法

```
fdformat [-n] device
```

参数：

- -n 关闭确认功能。这个选项会关闭格式化之后的确认步骤。

实例

```
fdformat -n /dev/fd0h1440
```

将磁碟机 A 的磁片格式化成 1.4MB 的磁片。并且省略确认的步骤。

Linux hdparm命令

Linux hdparm命令用于显示与设定硬盘的参数。

hdparm可检测，显示与设定IDE或SCSI硬盘的参数。

语法

```
hdparm [-CfghiIqtTvyYZ][-a <快取分区>][-A <0或1>][-c <I/O模式>][-d <0或1>][-k <0或1>][-K <0或
```

参数说明：

- -a<快取分区> 设定读取文件时，预先存入块区的分区数，若不加上<快取分区>选项，则显示目前的设定。
- -A<0或1> 启动或关闭读取文件时的快取功能。
- -c<I/O模式> 设定IDE32位I/O模式。
- -C 检测IDE硬盘的电源管理模式。
- -d<0或1> 设定磁盘的DMA模式。
- -f 将内存缓冲区的数据写入硬盘，并清楚缓冲区。
- -g 显示硬盘的磁轨，磁头，磁区等参数。
- -h 显示帮助。
- -i 显示硬盘的硬件规格信息，这些信息是在开机时由硬盘本身所提供。
- -I 直接读取硬盘所提供的硬件规格信息。
- -k<0或1> 重设硬盘时，保留-dmu参数的设定。
- -K<0或1> 重设硬盘时，保留-APSWXZ参数的设定。
- -m<磁区数> 设定硬盘多重分区存取的分区数。
- -n<0或1> 忽略硬盘写入时所发生的错误。
- -p<PIO模式> 设定硬盘的PIO模式。
- -P<磁区数> 设定硬盘内部快取的分区数。
- -q 在执行后续的参数时，不在屏幕上显示任何信息。
- -r<0或1> 设定硬盘的读写模式。
- -S<时间> 设定硬盘进入省电模式前的等待时间。
- -t 评估硬盘的读取效率。
- -T 平谷硬盘快取的读取效率。
- -u<0或1> 在硬盘存取时，允许其他中断要求同时执行。
- -v 显示硬盘的相关设定。
- -W<0或1> 设定硬盘的写入快取。
- -X<传输模式> 设定硬盘的传输模式。

- -y 使IDE硬盘进入省电模式。
- -Y 使IDE硬盘进入睡眠模式。
- -Z 关闭某些Seagate硬盘的自动省电功能。

实例

显示硬盘的相关设置：

```
# hdparm /dev/sda
 /dev/sda:
 IO_support = 0 (default 16-bit)
 readonly = 0 (off)
 readahead = 256 (on)
 geometry = 19929［柱面数］/255［磁头数］/63［扇区数］, sectors = 320173056［总扇区数］, start =
```

显示硬盘的柱面、磁头、扇区数

```
# hdparm -g /dev/sda
 /dev/sda:
 geometry = 19929［柱面数］/255［磁头数］/63［扇区数］, sectors = 320173056［总扇区数］, start =
```

评估硬盘的读取效率

```
 hdparm -t /dev/sda
 /dev/sda:
 Timing buffered disk reads: 166 MB in 3.03 seconds = 54.85 MB/sec
 [root@linuxso.com ~]# hdparm -t /dev/sda
 /dev/sda:
 Timing buffered disk reads: 160 MB in 3.01 seconds = 53.11 MB/sec
 [root@linuxso.com ~]# hdparm -t /dev/sda
 /dev/sda:
 Timing buffered disk reads: 166 MB in 3.00 seconds = 55.31 MB/sec
```

Linux mformat命令

Linux mformat命令用于对MS-DOS文件系统的磁盘进行格式化。

在已经做过低阶格式化的磁片上建立 DOS 档案系统。如果在编程 mtools 的时候把 USE_2M 的参数打开，部分与 2M 格式相关的参数就会发生作用。否则这些参数（像是 S,2,1,M）不会发生作用。

语法

```
mformat [-t cylinders] [-h heads] [-s sectors] [-l volume_label] [-F] [-I fsVer-sion] [-S
```

参数：

- -t 磁柱（synlider）数
- -h 磁头（head）数
- -s 每一磁轨的磁区数
- -l 标签
- -F 将磁碟格式化为 FAT32 格式，不过这个参数还在实验中。
- -I 设定 FAT32 中的版本号。这当然也还在实验中。
- -S 磁区大小代码，计算方式为 sector = 2^(大小代码+7)
- -c 磁丛（cluster）的磁区数。如果所给定的数字会导致磁丛数超过 FAT 表的限制，mformat 会自动放大磁区数。
- -s
- -M 软件磁区大小。这个数字就是系统回报的磁区大小。通常是和实际的大小相同。
- -a 如果加上这个参数，mformat 会产生一组 Atari 系统的序号给这块软碟。
- -X 将软碟格式化成 XDF 格式。使用前必须先用 xdfcopy 指令对软碟作低阶格式化的动作。
- -C 产生一个可以安装 MS-DOS 档案系统的磁碟影像档（disk image）。当然对一个实体磁碟机下这个参数是没有意义的。
- -H 隐藏磁区的数目。这通常适用在格式化硬盘的分割区时，因为通常一个分割区的前面还有分割表。这个参数未经测试，能不用就不用。
- -n 磁碟序号
- -r 根目录的大小，单位是磁区数。这个参数只对 FAT12 和 FAT16 有效。
- -B 使用所指定的档案或是设备的开机磁区做为这片磁片或分割区的开机磁区。当然当中的硬件参数会随之更动。
- -k 尽量保持原有的开机磁区。
- -0 第 0 轨的资料传输率

- -A 第 0 轨以外的资料传输率
- -2 使用 2m 格式
- -1 不使用 2m 格式

实例

用预设值把 a: （就是 /dev/fd0）里的磁碟片格式化。

```
mformat a:
```

Linux mkdosfs命令

Linux mkdosfs命令用于建立DOS文件系统。

device 指你想要建立 DOS 档案系统的装置代号。像是 /dev/hda1 等等。 block_count 则是你希望配置的区块数。如果 block_count 没有指定则系统会自动替你计算符合该装置大小的区块数。

```
mkdosfs [ -c | -l filename ]
     [ -f number_of_FATs ]
     [ -F FAT_size ]
     [ -i volume_id ]
     [ -m message_file ]
     [ -n volume_name ]
     [ -r root_dir_entry ]
     [ -s sector_per_cluster ]
     [ -v ]
     device
     [ block_count ]
```

参数：

- -c 建立档案系统之前先检查是否有坏轨。
- -l 从得定的档案中读取坏轨记录。
- -f 指定档案配置表（FAT , File Allocation Table)的数量。预设值为 2 。目前 Linux 的 FAT 档案系统不支援超过 2 个 FAT 表。通常这个不需要改。
- -F 指定 FAT 表的大小，通常是 12 或是 16 个位元组。12 位元组通常用于磁碟片，16 位元组用于一般硬盘的分割区，也就是所谓的 FAT16 格式。这个值通常系统会自己选定适当的值。在磁碟片上用 FAT16 通常不会发生作用，反之在硬盘上用 FAT12 亦然。
- -i 指定 Volume ID。一般是一个 4 个位元组的数字，像是 2e203a47 。如果不给系统会自己产生。
- -m 当使用者试图用这片磁片或是分割区开机，而上面没有操作系统时，系统会给使用者一段警告讯息。这个参数就是用来变更这个讯息的。你可以先用档案编辑好，然后用这个参数指定，或是用
- -m -
- 这样系统会要求你直接输入这段文字。要特别注意的是，档案里的字串长度不要超过 418 个字，包括展开的跳栏符号（TAB）和换行符号（换行符号在 DOS 底下算两个字元！）
- -n 指定 Volume Name，就是磁碟标签。如同在 DOS 底下的 format 指令一样，给不给都可以。没有预设值。
- -r 指定根目录底下的最大档案数。这里所谓的档案数包括目录。预设值是在软碟上是 112 或是 224 ，在硬盘上是 512。没事不要改这个数字。
- -s 每一个磁丛（cluster）的磁区数。必须是 2 的次方数。不过除非你知道你在作什么，这个值不要乱给。

- -v 提供额外的讯息

实例

将 A 槽里的磁碟片格式化为 DOS 格式，并将标签设为 Tester

```
mkdosfs -n Tester /dev/fd0
```

Linux mke2fs命令

Linux mke2fs命令用于建立ext2文件系统。

语法

```
mke2fs [-cFMqrSvV][-b <区块大小>][-f <不连续区段大小>][-i <字节>][-N <inode数>][-l <文件>][-L 
```

参数：

- -b<区块大小> 指定区块大小，单位为字节。
- -c 检查是否有损坏的区块。
- -f<不连续区段大小> 指定不连续区段的大小，单位为字节。
- -F 不管指定的设备为何，强制执行mke2fs。
- -i<字节> 指定"字节/inode"的比例。
- -N<inode数> 指定要建立的inode数目。
- -l<文件> 从指定的文件中，读取文件西中损坏区块的信息。
- -L<标签> 设置文件系统的标签名称。
- -m<百分比值> 指定给管理员保留区块的比例，预设为5%。
- -M 记录最后一次挂入的目录。
- -q 执行时不显示任何信息。
- -r 指定要建立的ext2文件系统版本。
- -R=<区块数> 设置磁盘阵列参数。
- -S 仅写入superblock与group descriptors，而不更改inode able inode bitmap以及block bitmap。
- -v 执行时显示详细信息。
- -V 显示版本信息。

Linux mkfs.ext2命令

功能说明：与 mke2fs命令 相同

Linux mkfs.msdos命令

功能说明：与 mkdosfs 命令 相同。

Linux mkinitrd命令

Linux mkinitrd命令用于建立要载入ramdisk的映像文件。

mkinitrd可建立映像文件，以供Linux开机时载入ramdisk。

语法

```
mkinitrd [-fv][--omit-scsi-modules][--version][--preload=<模块名称>][--with=<模块名称>][映像
```

参数：

- -f 若指定的映像问家名称与现有文件重复，则覆盖现有的文件。
- -v执行时显示详细的信息。
- --omit-scsi-modules 不要载入SCSI模块。
- --preload=<模块名称> 指定要载入的模块。
- --with=<模块名称> 指定要载入的模块。
- --version 显示版本信息。

Linux mkisofs命令

Linux mkisofs命令用于建立ISO 9660映像文件。

mkisofs可将指定的目录与文件做成ISO 9660格式的映像文件，以供刻录光盘。

语法

```
mkisofs [-adDfhJlLNrRTvz][-print-size][-quiet][-A <应用程序ID>][-abstract <摘要文件>][-b <开
```

参数：

- -a或--all mkisofs通常不处理备份文件。使用此参数可以把备份文件加到映像文件中。
- -A<应用程序ID>或-appid<应用程序ID> 指定光盘的应用程序ID。
- -abstract<摘要文件> 指定摘要文件的文件名。
- -b<开机映像文件>或-eltorito-boot<开机映像文件> 指定在制作可开机光盘时所需的开机映像文件。
- -biblio<ISBN文件> 指定ISBN文件的文件名，ISBN文件位于光盘根目录下，记录光盘的ISBN。
- -c<开机文件名称> 制作可开机光盘时，mkisofs会将开机映像文件中的全-eltorito-catalog<开机文件名称>全部内容作成一个文件。
- -C<盘区编号，盘区编号> 将许多节区合成一个映像文件时，必须使用此参数。
- -copyright<版权信息文件> 指定版权信息文件的文件名。
- -d或-omit-period 省略文件后的句号。
- -D或-disable-deep-relocation ISO 9660最多只能处理8层的目录，超过8层的部分，RRIP会自动将它们设置成ISO 9660兼容的格式。使用-D参数可关闭此功能。
- -f或-follow-links 忽略符号连接。
- -h 显示帮助。
- -hide<目录或文件名> 使指定的目录或文件在ISO 9660或Rock RidgeExtensions的系统中隐藏。
- -hide-joliet<目录或文件名> 使指定的目录或文件在Joliet系统中隐藏。
- -J或-joliet 使用Joliet格式的目录与文件名称。
- -l或-full-iso9660-filenames 使用ISO 9660 32字符长度的文件名。
- -L或-allow-leading-dots 允许文件名的第一个字符为句号。
- -log-file<记录文件> 在执行过程中若有错误信息，预设会显示在屏幕上。
- -m<目录或文件名>或-exclude<目录或文件名> 指定的目录或文件名将不会房入映像文件中。
- -M<映像文件>或-prev-session<映像文件> 与指定的映像文件合并。

- -N或-omit-version-number 省略ISO 9660文件中的版本信息。
- -o<映像文件>或-output<映像文件> 指定映像文件的名称。
- -p<数据处理人>或-preparer<数据处理人> 记录光盘的数据处理人。
- -print-size 显示预估的文件系统大小。
- -quiet 执行时不显示任何信息。
- -r或-rational-rock 使用Rock Ridge Extensions，并开放全部文件的读取权限。
- -R或-rock 使用Rock Ridge Extensions。
- -sysid<系统ID> 指定光盘的系统ID。
- -T或-translation-table 建立文件名的转换表，适用于不支持Rock Ridge Extensions的系统上。
- -v或-verbose 执行时显示详细的信息。
- -V<光盘ID>或-volid<光盘ID> 指定光盘的卷册集ID。
- -volset-size<光盘总数> 指定卷册集所包含的光盘张数。
- -volset-seqno<卷册序号> 指定光盘片在卷册集中的编号。
- -x<目录> 指定的目录将不会放入映像文件中。
- -z 建立通透性压缩文件的SUSP记录，此记录目前只在Alpha机器上的Linux有效。

Linux mkswap命令

Linux mkswap命令用于设置交换区(swap area)。

mkswap可将磁盘分区或文件设为Linux的交换区。

语法

```
mkswap [-cf][-v0][-v1][设备名称或文件][交换区大小]
```

参数：

- -c 建立交换区前，先检查是否有损坏的区块。
- -f 在SPARC电脑上建立交换区时，要加上此参数。
- -v0 建立旧式交换区，此为预设值。
- -v1 建立新式交换区。
- [交换区大小] 指定交换区的大小，单位为1024字节。

Linux mpartition命令

Linux mpartition命令用于建立或删除MS-DOS的分区。

mpartition为mtools工具指令，可建立或删除磁盘分区。

语法

```
mpartition [-acdfIprv][-b <磁区数>][-h <磁头数>][l <磁区数>][-s <磁区数>][-t <柱面数>][驱动器代
```

参数：

- -a 将分区设置为可开机分区。
- -b<磁区数> 建立分区时，指定要从第几个磁区开始建立分区。
- -c 建立分区。
- -d 将分区设置为无法开机的分区。
- -f 强制地修改分区而不管检查时发生的错误信息。
- -h<磁头数> 建立分区时，指定分区的磁头数。
- -I 删除全部的分区。
- -l<磁区数> 建立分区时，指定分区的容量大小，单位为磁区数。
- -p 当要重新建立分区时，显示命令列。
- -r 删除分区。
- -s<磁区数> 建立分区时，指定每个磁轨的磁区数。
- -t<柱面数> 建立分区时，指定分区的柱面数。
- -v 与-p参数一并使用，若没有同时下达修改分区的命令，则显示目前分区的状态。

Linux swapon命令

Linux swapon命令用于激活Linux系统中交换空间，Linux系统的内存管理必须使用交换区来建立虚拟内存。

语法

```
/sbin/swapon -a [-v]
/sbin/swapon [-v] [-p priority] specialfile ...
/sbin/swapon [-s]
```

参数说明：

- -h 请帮帮我
- -V 显示版本讯息
- -s 显示简短的装置讯息
- -a 自动启动所有SWAP装置
- -p 设定优先权，你可以在0到32767中间选一个数字给他。或是在 /etc/fstab 里面加上 pri=[value] ([value]就是0~32767中间一个数字)，然后你就可以很方便的直接使用 swapon -a 来启动他们，而且有优先权设定。

swapon 是开启swap.

相对的,便有一个关闭swap的指令,swapoff.

Linux symlinks命令

Linux symlinks命令用于维护符号连接的工具程序。

symlinks可检查目录中的符号连接，并显示符号连接类型。以下为symlinks可判断的符号连接类型：

- absolute：符号连接使用了绝对路径。
- dangling：原始文件已经不存在。
- lengthy：符号连接的路径中包含了多余的"../"。
- messy：符号连接的路径中包含了多余的"/"。
- other_fs：原始文件位于其他文件系统中。
- relative：符号连接使用了相对路径。

语法

```
symlinks [-cdrstv][目录]
```

参数：

- -c 将使用绝对路径的符号连接转换为相对路径。
- -d 移除dangling类型的符号连接。
- -r 检查目录下所有子目录中的符号连接。
- -s 检查lengthy类型的符号连接。
- -t 与-c一并使用时，会显示如何将绝对路径的符号连接转换为相对路径，但不会实际转换。
- -v 显示所有类型的符号连接。

Linux sync命令

Linux sync命令用于数据同步,sync命令是在关闭Linux系统时使用的。

Linux 系统中欲写入硬盘的资料有的时候会了效率起见，会写到 filesystem buffer 中，这个 buffer 是一块记忆体空间，如果欲写入硬盘的资料存于此 buffer 中，而系统又突然断电的话，那么资料就会流失了，sync 指令会将存于 buffer 中的资料强制写入硬盘中。

语法

```
sync
```

Linux mbadblocks命令

Linux mbadblocks命令用于检查MS-DOS文件系统的磁盘是否有损坏的磁区。

mbadblocks为mtools工具指令，可用来扫描MS-DOS文件系统的磁盘驱动器，并标示出损坏的磁区。

语法

```
mbadblocks [驱动器代号]
```

Linux mkfs.minix命令

Linux mkfs.minix命令用于建立Minix文件系统。

mkfs.minix可建立Minix文件系统。

语法

```
mkfs.minix [-cv][-i <inode数目>][-l <文件>][-n <文件名长度>][设备名称][区块数]
```

参数：

- -c 检查是否有损坏的区块。
- -i<inode数目> 指定文件系统的inode总数。
- -l<文件> 从指定的文件中，读取文件系统中损坏区块的信息。
- -n<文件名长度> 指定文件名称长度的上限。
- -v 建立第2版的Minix文件系统。

Linux fsck.ext2命令

Linux fsck.ext2命令用于检查文件系统并尝试修复错误。

当ext2文件系统发生错误时，可用fsck.ext2指令尝试加以修复。

语法

```
fsck.ext2 [-acdfFnprsStvVy][-b <分区第一个磁区地址>][-B <区块大小>][-C <反叙述器>][-I <inode缓冲
```

参数：

- -a 自动修复文件系统，不询问任何问题。
- -b<分区第一个磁区地址> 指定分区的第一个磁区的起始地址，也就是Super Block。
- -B<区块大小> 设置该分区每个区块的大小。
- -c 检查指定的文件系统内，是否存在有损坏的区块。
- -C<反叙述器> 指定反叙述器，fsck.ext2指令会把全部的执行过程，都交由其逆向叙述，便于排错或监控程序执行的情形。
- -d 详细显示指令执行过程，便于排错或分析程序执行的情形。
- -f 强制对该文件系统进行完整检查，纵然该文件系统在慨略检查下没有问题。
- -F 检查文件系统之前，先清理该保存设备块区内的数据。
- -I<inode缓冲区块数> 设置欲检查的文件系统，其inode缓冲区的区块数目。
- -l<损坏区块文件> 把文件中所列出的区块，视为损坏区块并将其标示出来，避免应用程序使用该区块。
- -L<损坏区块文件> 此参数的效果和指定"-l"参数类似，但在参考损坏区块文件标示损坏区块之前，会先将原来标示成损坏区块者统统清楚，即全部重新设置，而非仅是加入新的损坏区块标示。
- -n 把欲检查的文件系统设成只读，并关闭互动模式，否决所有询问的问题。
- -p 此参数的效果和指定"-a"参数相同。
- -P<处理inode大小> 设置fsck.ext2指令所能处理的inode大小为多少。
- -r 此参数将忽略不予处理，仅负责解决兼容性的问题。
- -s 检查文件系统时，交换每对字节的内容。
- -S 此参数的效果和指定"-s"参数类似，但不论该文件系统是否已是标准位顺序，一律交换每对字节的内容。
- -t 显示fsck.ext2指令的时序信息。
- -v 详细显示指令执行过程。
- -V 显示版本信息。
- -y 关闭互动模式，且同意所有询问的问题。

Linux fdisk命令

Linux fdisk是一个创建和维护分区表的程序，它兼容DOS类型的分区表、BSD或者SUN类型的磁盘列表。

语法

```
fdisk [必要参数][选择参数]
```

必要参数：

- -l 列出素所有分区表
- -u 与"-l"搭配使用，显示分区数目

选择参数：

- -s<分区编号> 指定分区
- -v 版本信息

菜单操作说明

- m ：显示菜单和帮助信息
- a ：活动分区标记/引导分区
- d ：删除分区
- l ：显示分区类型
- n ：新建分区
- p ：显示分区信息
- q ：退出不保存
- t ：设置分区号
- v ：进行分区检查
- w ：保存修改
- x ：扩展应用，高级功能

实例

显示当前分区情况：

```
# fdisk -l

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

  Device Boot   Start     End   Blocks  Id System
/dev/sda1  *      1     13   104391  83 Linux
/dev/sda2       14    1305  10377990  8e Linux LVM

Disk /dev/sdb: 5368 MB, 5368709120 bytes
255 heads, 63 sectors/track, 652 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

Disk /dev/sdb doesn't contain a valid partition table
```

显示SCSI硬盘的每个分区情况

```
# fdisk -lu

Disk /dev/sda: 10.7 GB, 10737418240 bytes
255 heads, 63 sectors/track, 1305 cylinders, total 20971520 sectors
Units = sectors of 1 * 512 = 512 bytes

  Device Boot   Start     End   Blocks  Id System
/dev/sda1  *     63   208844   104391  83 Linux
/dev/sda2     208845  20964824  10377990  8e Linux LVM

Disk /dev/sdb: 5368 MB, 5368709120 bytes
255 heads, 63 sectors/track, 652 cylinders, total 10485760 sectors
Units = sectors of 1 * 512 = 512 bytes

Disk /dev/sdb doesn't contain a valid partition table
```

Linux losetup命令

Linux losetup命令用于设置循环设备。

循环设备可把文件虚拟成区块设备，籍以模拟整个文件系统，让用户得以将其视为硬盘驱动器，光驱或软驱等设备，并挂入当作目录来使用。

语法

```
losetup [-d][-e <加密方式>][-o <平移数目>][循环设备代号][文件]
```

参数：

- -d 卸除设备。
- -e<加密方式> 启动加密编码。
- -o<平移数目> 设置数据平移的数目。

实例

（1）创建空的磁盘镜像文件，这里创建一个1.44M的软盘

```
$ dd if=/dev/zero of=floppy.img bs=512 count=2880
```

（2）使用 losetup将磁盘镜像文件虚拟成快设备

```
$ losetup /dev/loop1 floppy.img
```

（3）挂载块设备

```
$ mount /dev/loop0 /tmp
```

经过上面的三步之后，我们就可以通过/tmp目录，像访问真实快设备一样来访问磁盘镜像文件floppy.img。

（4） 卸载loop设备

```
$ umount /tmp
$ losetup -d /dev/loop1
```

Linux mkfs命令

使用方式 : mkfs [-V] [-t fstype] [fs-options] filesys [blocks]

Linux mkfs命令用于在特定的分区上建立 linux 文件系统

参数 ：

- device ： 预备检查的硬盘分区，例如：/dev/sda1
- -V : 详细显示模式
- -t : 给定档案系统的型式，Linux 的预设值为 ext2
- -c : 在制做档案系统前，检查该partition 是否有坏轨
- -l bad_blocks_file : 将有坏轨的block资料加到 bad_blocks_file 里面
- block : 给定 block 的大小

实例

在 /dev/hda5 上建一个 msdos 的档案系统，同时检查是否有坏轨存在，并且将过程详细列出来 :

```
mkfs -V -t msdos -c /dev/hda5
```

将sda6分区格式化为ext3格式

```
mfks -t ext3 /dev/sda6
```

注意：这里的文件系统是要指定的，比如 ext3 ；reiserfs ；ext2 ；fat32 ；msdos 等。

Linux getty命令

Linux getty命令用于设置终端机模式，连线速率和管制线路。

getty指令是UNIX之类操作系统启动时所必须的3个步骤之一。

语法

```
getty [-h][-d<组态配置文件>][-r<延迟秒数>][-t<超时秒数>][-w<等待字符串>][终端机编号][连线速率<终端
```

参数：

- -c<定义配置文件> 指定定义配置文件，预设为/etc/gettydefs。
- -d<组态配置文件> 指定组态配置文件，预设为/etc/conf.getty。
- -h 当传输速率为0时就强制断线。
- -r<延迟秒数> 设置延迟时间。
- -t<超时秒数> 设置等待登入的时间。
- -w<等待字符串> 设置等待回应的字符串。

实例

开启终端：

```
# getty tty7
```

Linux sfdisk命令

Linux sfdisk命令是硬盘分区工具程序。

sfdisk为硬盘分区工具程序，可显示分区的设置信息，并检查分区是否正常。

语法

```
sfdisk [-?Tvx][-d <硬盘>][-g <硬盘>][-l <硬盘>][-s <分区>][-V <硬盘>]
```

参数：

- -?或--help 显示帮助。
- -d<硬盘> 显示硬盘分区的设置。
- -g<硬盘>或--show-geometry<硬盘> 显示硬盘的CHS参数。
- -l<硬盘> 显示后硬盘分区的相关设置。
- -s<分区> 显示分区的大小，单位为区块。
- -T或--list-types 显示所有sfdisk能辨识的文件系统ID。
- -v或--version 显示版本信息。
- -V<硬盘>或--verify<硬盘> 检查硬盘分区是否正常。
- -x或--show-extend 显示扩展分区中的逻辑分区。

实例

显示分区信息：

```
# sfdisk -l

Disk /dev/sda: 1305 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0

Device Boot Start End #cyls #blocks Id System
/dev/sda1 * 0+ 12 13- 104391 83 Linux
/dev/sda2 13 1304 1292 10377990 8e Linux LVM
/dev/sda3 0 - 0 0 0 Empty
/dev/sda4 0 - 0 0 0 Empty

Disk /dev/sdb: 652 cylinders, 255 heads, 63 sectors/track

sfdisk: ERROR: sector 0 does not have an msdos signature
/dev/sdb: unrecognized partition
No partitions found
```

Linux swapoff命令

Linux swapoff命令用于关闭系统交换区(swap area)。

swapoff实际上为swapon的符号连接，可用来关闭系统的交换区。

语法

```
swapoff [设备]
```

参数：

- -a 将/etc/fstab文件中所有设置为swap的设备关闭
- -h 帮助信息
- -V 版本信息

实例

显示分区信息:

```
# sfdisk -l //显示分区信息

Disk /dev/sda: 1305 cylinders, 255 heads, 63 sectors/track
Units = cylinders of 8225280 bytes, blocks of 1024 bytes, counting from 0

  Device Boot Start   End  #cyls  #blocks  Id System
/dev/sda1  *    0+   12   13-  104391  83 Linux
/dev/sda2     13  1304  1292  10377990  8e Linux LVM
/dev/sda3     0    -    0     0  0 Empty
/dev/sda4     0    -    0     0  0 Empty

Disk /dev/sdb: 652 cylinders, 255 heads, 63 sectors/track

sfdisk: ERROR: sector 0 does not have an msdos signature
/dev/sdb: unrecognized partition
No partitions found
```

关闭交换分区。

```
# swapoff /dev/sda2 // 关闭交换分区
```

Linux命令大全 - 网络通讯

apachectl	arpwatch	dip	getty
mingetty	uux	telnet	uulog
uustat	ppp-off	netconfig	nc
httpd	ifconfig	minicom	mesg
dnsconf	wall	netstat	ping
pppstats	samba	setserial	talk
traceroute	tty	newaliases	uuname
netconf	write	statserial	efax
pppsetup	tcpdump	ytalk	cu
smbd	testparm	smbclient	shapecfg

Linux apachectl命令

Linux apachectl命令可用来控制Apache HTTP服务器的程序。

apachectl是slackware内附Apache HTTP服务器的script文件，可供管理员控制服务器，但在其他Linux的Apache HTTP服务器不一定有这个文件。

语法

```
apachectl [configtest][fullstatus][graceful][help][restart][start][status][stop]
```

参数：

- configtest 检查设置文件中的语法是否正确。
- fullstatus 显示服务器完整的状态信息。
- graceful 重新启动Apache服务器，但不会中断原有的连接。
- help 显示帮助信息。
- restart 重新启动Apache服务器。
- start 启动Apache服务器。
- status 显示服务器摘要的状态信息。
- stop 停止Apache服务器。

Linux arpwatch命令

Linux arpwatch命令用于监听网络上ARP的记录。

ARP(Address Resolution Protocol)是用来解析IP与网络装置硬件地址的协议。

arpwatch可监听区域网络中的ARP数据包并记录，同时将监听到的变化通过E-mail来报告。

语法

```
arpwatch [-d][-f<记录文件>][-i<接口>][-r<记录文件>]
```

参数：

- -d 启动排错模式。
- -f<记录文件> 设置存储ARP记录的文件，预设为/var/arpwatch/arp.dat。
- -i<接口> 指定监听ARP的接口，预设的接口为eth0。
- -r<记录文件> 从指定的文件中读取ARP记录，而不是从网络上监听。
- -n 指定附加的本地网络
- -u 指定用户和用户组
- -e 发送邮件给指定用户，非默认的root用户
- -s 指定用户名作为返回地址，而不是默认的用户root

实例

监听网卡eth0的ARP信息

```
arpwatch -i eth0
```

监听ARP的信息，将相关信息记录到相应的文件

```
# arpwatch -i eth0 -f a.log //将信息记录到a.log中
```

Linux nc命令

Linux nc命令用于设置路由器。

执行本指令可设置路由器的相关参数。

语法

```
nc [-hlnruz][-g<网关...>][-G<指向器数目>][-i<延迟秒数>][-o<输出文件>][-p<通信端口>][-s<来源位址>]
```

参数说明：

- -g<网关> 设置路由器跃程通信网关，最丢哦可设置8个。
- -G<指向器数目> 设置来源路由指向器，其数值为4的倍数。
- -h 在线帮助。
- -i<延迟秒数> 设置时间间隔，以便传送信息及扫描通信端口。
- -l 使用监听模式，管控传入的资料。
- -n 直接使用IP地址，而不通过域名服务器。
- -o<输出文件> 指定文件名称，把往来传输的数据以16进制字码倾倒成该文件保存。
- -p<通信端口> 设置本地主机使用的通信端口。
- -r 乱数指定本地与远端主机的通信端口。
- -s<来源位址> 设置本地主机送出数据包的IP地址。
- -u 使用UDP传输协议。
- -v 显示指令执行过程。
- -w<超时秒数> 设置等待连线的时间。
- -z 使用0输入/输出模式，只在扫描通信端口时使用。

实例

TCP端口扫描

```
# nc -v -z -w2 192.168.0.3 1-100
192.168.0.3: inverse host lookup failed: Unknown host
(UNKNOWN) [192.168.0.3] 80 (http) open
(UNKNOWN) [192.168.0.3] 23 (telnet) open
(UNKNOWN) [192.168.0.3] 22 (ssh) open
```

扫描192.168.0.3 的端口 范围是 1-100

扫描UDP端口

```
# nc -u -z -w2 192.168.0.1 1-1000 //扫描192.168.0.3 的端口 范围是 1-1000
```

扫描指定端口

```
# nc -nvv 192.168.0.1 80 //扫描 80端口
(UNKNOWN) [192.168.0.1] 80 (?) open
y  //用户输入
```

Linux dip命令

Linux dip命令用于IP拨号连接。

dip可控制调制解调器，以拨号IP的方式建立对外的双向连接。

语法

```
dip [-aikltv][-m<MTU数目>][-p<协议>][拨号script文件]
```

参数说明：

- -a 询问用户名称与密码。
- -i 启动拨号服务器功能。
- -k 删除执行中的dip程序。
- -l 指定要删除的连线，必须配合-k参数一起使用。
- -m<MTU数目> 设置最大传输单位，预设值为296。
- -p<协议> 设置通信协议。
- -t 进入dip的指令模式。
- -v 执行时显示详细的信息。

实例

建立拨号连接

```
$ dip -t
```

Linux mingetty命令

Linux mingetty命令是精简版的getty。

mingetty适用于本机上的登入程序。

语法

```
mingetty [--long-hostname][--noclear][tty]
```

参数说明：

- --long-hostname 显示完整的主机名称。
- --noclear 在询问登入的用户名称之前不要清楚屏幕画面。

Linux netconfig命令

Linux netconfig命令用于设置网络环境。

这是Slackware发行版内附程序，它具有互动式的问答界面，让用户轻易完成网络环境的设置。

语法

```
netconfig
```

Linux ppp-off命令

Linux ppp命令用于关闭ppp连线。

这是Slackware发行版内附的程序，让用户切断PPP的网络连线。

语法

```
ppp-off
```

实例

关闭ppp连线

```
# ppp-off
```

Linux uustat命令

Linux uustat命令用于显示UUCP目前的状况。

执行uucp与uux指令后，会先将工作送到队列，再由uucico来执行工作。uustat可显示，删除或启动队列中等待执行的工作。

语法

```
uustat [-aeiKmMNpqQRv][-B<行数>][-c<指令>][-C<指令>][-I<配置文件>][-k<工作>][-o<小时>][-r<工作
```

参数说明：

- -a或-all 显示全部的UUCP工作。
- -B<行数>或--mail-lines<行数> 与-M或-N参数一并使用，用来指定邮件中要包含多少行的信息。
- -c<指令>或--command<指令> 显示与<指令>有关的工作。
- -C<指令>或--not-command<指令> 显示与<指令>无关的工作。
- -e或--executions 仅显示待执行的工作。
- -i或--prompt 针对队列中的每项工作，询问使用是否要删除工作。
- -I<配置文件>或--config<配置文件> 指定配置文件。
- -k<工作>或--kill<工作> 删除指定的工作。
- -m或--status 删除全部的工作。
- -M或-mail 将状态信息邮寄给UUCP管理员。
- -N或--notify 将状态信息分别邮寄给提出该项工作的用户。
- -o<小时>或--older-than<小时> 显示超过指定时数的工作。
- -p或--ps 显示负责UUCP锁定的程序。
- -q或--list 显示每台远端主机上所要执行工作的状态。
- -Q或--no-list 不显示工作。
- -r<工作>或--rejuvenate<工作> 重新启动指定的工作。
- -R或--rejuvenate-all 重新启动全部的工作。
- -s<主机>或--system<主机> 显示与<主机>有关的工作。
- -S<主机>或--not-system<主机> 显示与<主机>无关的工作。
- -v或--version 显示版本信息。
- -u<用户>或--user<用户> 显示与<用户>有关的工作。
- -U<用户>或--not-user<用户> 显示与<用户>无关的工作。
- -W<附注>或--comment<附注> 要放在邮件信息中的附注。
- -y<小时>或--younger-than<小时> 显示低于指定时数的工作。

- -x<层级>或--debug<层级> 指定排错层级。
- --help 显示帮助。

实例

显示所有任务

```
# uustat -a
```

显示等待的任务

```
# uustat -e
```

Linux uulog命令

Linux uulog命令用于显示UUCP记录文件。

uulog可用来显示UUCP记录文件中记录。

语法

```
uulog [-DFISv][-<行数>][-f<主机>][-I<配置文件>][-n<行数>][-s<主机>][-u<用户>][-X<层级>][--help
```

参数说明：

- -D或--debuglog 显示排错记录。
- -f<主机>或--follow<主机> 与-F参数类似，但仅显示与指定主机相关的记录。
- -I<配置文件>或--config<配置文件> 指定程序的配置文件。
- -<行数>,-n<行数>或--lines<行数> 显示记录文件中，从最后算起指定行数的数值。
- -s<主机> 仅显示记录文件中，与指定文件相关的记录。
- -S或--statslog 显示统计记录。
- -u<用户>或--suer<用户> 仅显示记录文件中，与指定用户相关的记录。
- -v或--version 显示版本信息。
- -X<层级>或--debug<层级> 设定排错层级。
- --help 显示帮助。

实例

显示uucp log信息

```
# uulog
```

Linux wall命令

Linux wall命令会将讯息传给每一个 mesg 设定为 yes 的上线使用者。当使用终端机介面做为标准传入时, 讯息结束时需加上 EOF (通常用 Ctrl+D)。

使用权限：所有使用者。

语法

```
wall [ message ]
```

实例

传讯息"hi" 给每一个使用者

```
wall hi
```

广播消息

```
# wall Ilove

Broadcast message from root (pts/4) (Thu May 27 16:41:09 2014):

Ilove
```

Linux uux命令

Linux uux命令用于在远端的UUCP主机上执行指令。

uux可在远端的UUCP主机上执行指令或是执行本机上的指令，但在执行时会使用远端电脑的文件。

语法

```
uux [-bcCIjlnrvz][-a<地址>][-g<等级>][-s<文件>][-x<层级>][--help][指令]
```

参数说明：

- -或-p或--stdin 直接从键盘读取要执行的指令。
- -a<地址>或--requestor<地址> 执行邮件地址，以便寄送状态信息。
- -b或--erturn-stdin 在屏幕上显示状态信息。
- -c或--nocopy 不用将文件复制到缓冲区。
- -C或--copy 将文件复制到缓冲区。
- -g<等级>或--grade<等级> 指定文件传送作业的优先顺序。
- -I或--config file 指定uux配置文件。
- -j或--jobid 显示作业编号。
- -l或--link 将本机上的文件连接到缓冲区。
- -n或--notification=no 无论发生任何状态，都不寄邮件通知用户。
- -r或--nouucico 不要立即启动uucico服务程序，仅将作业送到队列中，然后再执行。
- -s<文件>或--status<文件> 将完成状态保存为指定的文件。
- -v或--version 显示版本信息。
- -x<层级>或--debug<层级> 指定排错层级。
- -z或--notification=error 若发生错误，则以邮件来通知用户。
- --help 显示帮助。

实例

在远程主机 uucp 执行命令

```
# uux hnlinux! date  /// 在远程主机 指定date命令查看系统时间
```

Linux telnet命令

Linux telnet命令用于远端登入。

执行telnet指令开启终端机阶段作业，并登入远端主机。

语法

```
telnet [-8acdEfFKLrx][-b<主机别名>][-e<脱离字符>][-k<域名>][-l<用户名称>][-n<记录文件>][-S<服务
```

参数说明：

- -8 允许使用8位字符资料，包括输入与输出。
- -a 尝试自动登入远端系统。
- -b<主机别名> 使用别名指定远端主机名称。
- -c 不读取用户专属目录里的.telnetrc文件。
- -d 启动排错模式。
- -e<脱离字符> 设置脱离字符。
- -E 滤除脱离字符。
- -f 此参数的效果和指定"-F"参数相同。
- -F 使用Kerberos V5认证时，加上此参数可把本地主机的认证数据上传到远端主机。
- -k<域名> 使用Kerberos认证时，加上此参数让远端主机采用指定的领域名，而非该主机的域名。
- -K 不自动登入远端主机。
- -l<用户名称> 指定要登入远端主机的用户名称。
- -L 允许输出8位字符资料。
- -n<记录文件> 指定文件记录相关信息。
- -r 使用类似rlogin指令的用户界面。
- -S<服务类型> 设置telnet连线所需的IP TOS信息。
- -x 假设主机有支持数据加密的功能，就使用它。
- -X<认证形态> 关闭指定的认证形态。

实例

登录远程主机

```
# telnet 192.168.0.5

//登录IP为 192.168.0.5 的远程主机
```

Linux netstat命令

Linux netstat命令用于显示网络状态。

利用netstat指令可让你得知整个Linux系统的网络情况。

语法

```
netstat [-acCeFghilMnNoprstuvVwx][-A<网络类型>][--ip]
```

参数说明：

- -a或--all 显示所有连线中的Socket。
- -A<网络类型>或--<网络类型> 列出该网络类型连线中的相关地址。
- -c或--continuous 持续列出网络状态。
- -C或--cache 显示路由器配置的快取信息。
- -e或--extend 显示网络其他相关信息。
- -F或--fib 显示FIB。
- -g或--groups 显示多重广播功能群组组员名单。
- -h或--help 在线帮助。
- -i或--interfaces 显示网络界面信息表单。
- -l或--listening 显示监控中的服务器的Socket。
- -M或--masquerade 显示伪装的网络连线。
- -n或--numeric 直接使用IP地址，而不通过域名服务器。
- -N或--netlink或--symbolic 显示网络硬件外围设备的符号连接名称。
- -o或--timers 显示计时器。
- -p或--programs 显示正在使用Socket的程序识别码和程序名称。
- -r或--route 显示Routing Table。
- -s或--statistice 显示网络工作信息统计表。
- -t或--tcp 显示TCP传输协议的连线状况。
- -u或--udp 显示UDP传输协议的连线状况。
- -v或--verbose 显示指令执行过程。
- -V或--version 显示版本信息。
- -w或--raw 显示RAW传输协议的连线状况。
- -x或--unix 此参数的效果和指定"-A unix"参数相同。
- --ip或--inet 此参数的效果和指定"-A inet"参数相同。

实例

显示详细的网络状况

```
# netstat -a
```

显示当前户籍UDP连接状况

```
# netstat -nu
```

显示UDP端口号的使用情况

```
# netstat -apu
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address         Foreign Address       State    PID/Program name
udp    0   0 *:32768            *:*                   -
udp    0   0 *:nfs            *:*                   -
udp    0   0 *:641            *:*                   3006/rpc.statd
udp    0   0 192.168.0.3:netbios-ns   *:*                   3537/nmbd
udp    0   0 *:netbios-ns        *:*                   3537/nmbd
udp    0   0 192.168.0.3:netbios-dgm   *:*                   3537/nmbd
udp    0   0 *:netbios-dgm        *:*                   3537/nmbd
udp    0   0 *:tftp            *:*                   3346/xinetd
udp    0   0 *:999            *:*                   3366/rpc.rquotad
udp    0   0 *:sunrpc           *:*                   2986/portmap
udp    0   0 *:ipp            *:*                   6938/cupsd
udp    0   0 *:1022            *:*                   3392/rpc.mountd
udp    0   0 *:638            *:*                   3006/rpc.statd
```

显示网卡列表

```
# netstat -i
Kernel Interface table
Iface     MTU Met  RX-OK RX-ERR RX-DRP RX-OVR  TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0     1500  0  181864   0    0    0  141278   0    0    0 BMRU
lo     16436  0   3362   0    0    0   3362   0    0    0 LRU
```

显示组播组的关系

```
# netstat -g
IPv6/IPv4 Group Memberships
Interface    RefCnt Group
--------------- ------ ---------------------
lo        1    ALL-SYSTEMS.MCAST.NET
eth0       1    ALL-SYSTEMS.MCAST.NET
lo        1    ff02::1
eth0       1    ff02::1:ff0a:b0c
eth0       1    ff02::1
```

显示网络统计信息

```
# netstat -s
Ip:
  184695 total packets received
  0 forwarded
  0 incoming packets discarded
```

```
    184687 incoming packets delivered
    143917 requests sent out
    32 outgoing packets dropped
    30 dropped because of missing route
Icmp:
    676 ICMP messages received
    5 input ICMP message failed.
    ICMP input histogram:
        destination unreachable: 44
        echo requests: 287
        echo replies: 345
    304 ICMP messages sent
    0 ICMP messages failed
    ICMP output histogram:
        destination unreachable: 17
        echo replies: 287
Tcp:
    473 active connections openings
    28 passive connection openings
    4 failed connection attempts
    11 connection resets received
    1 connections established
    178253 segments received
    137936 segments send out
    29 segments retransmited
    0 bad segments received.
    336 resets sent
Udp:
    5714 packets received
    8 packets to unknown port received.
    0 packet receive errors
    5419 packets sent
TcpExt:
    1 resets received for embryonic SYN_RECV sockets
    ArpFilter: 0
    12 TCP sockets finished time wait in fast timer
    572 delayed acks sent
    3 delayed acks further delayed because of locked socket
    13766 packets directly queued to recvmsg prequeue.
    1101482 packets directly received from backlog
    19599861 packets directly received from prequeue
    46860 packets header predicted
    14541 packets header predicted and directly queued to user
    TCPPureAcks: 12259
    TCPHPAcks: 9119
    TCPRenoRecovery: 0
    TCPSackRecovery: 0
    TCPSACKReneging: 0
    TCPFACKReorder: 0
    TCPSACKReorder: 0
    TCPRenoReorder: 0
    TCPTSReorder: 0
    TCPFullUndo: 0
    TCPPartialUndo: 0
    TCPDSACKUndo: 0
    TCPLossUndo: 0
    TCPLoss: 0
    TCPLostRetransmit: 0
    TCPRenoFailures: 0
    TCPSackFailures: 0
    TCPLossFailures: 0
    TCPFastRetrans: 0
    TCPForwardRetrans: 0
    TCPSlowStartRetrans: 0
    TCPTimeouts: 29
    TCPRenoRecoveryFail: 0
    TCPSackRecoveryFail: 0
    TCPSchedulerFailed: 0
    TCPRcvCollapsed: 0
    TCPDSACKOldSent: 0
    TCPDSACKOfoSent: 0
    TCPDSACKRecv: 0
```

```
  TCPDSACKOfoRecv: 0
  TCPAbortOnSyn: 0
  TCPAbortOnData: 1
  TCPAbortOnClose: 0
  TCPAbortOnMemory: 0
  TCPAbortOnTimeout: 3
  TCPAbortOnLinger: 0
  TCPAbortFailed: 3
  TCPMemoryPressures: 0
```

显示监听的套接口

```
# netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address        Foreign Address       State
tcp    0   0 *:32769           *:*              LISTEN
tcp    0   0 *:nfs            *:*              LISTEN
tcp    0   0 *:644            *:*              LISTEN
tcp    0   0 *:1002           *:*              LISTEN
tcp    0   0 *:netbios-ssn        *:*              LISTEN
tcp    0   0 *:sunrpc          *:*              LISTEN
tcp    0   0 vm-dev:ipp         *:*              LISTEN
tcp    0   0 *:telnet          *:*              LISTEN
tcp    0   0 *:601            *:*              LISTEN
tcp    0   0 *:microsoft-ds       *:*              LISTEN
tcp    0   0 *:http           *:*              LISTEN
tcp    0   0 *:ssh            *:*              LISTEN
tcp    0   0 *:https           *:*              LISTEN
udp    0   0 *:32768           *:*
udp    0   0 *:nfs            *:*
udp    0   0 *:641            *:*
udp    0   0 192.168.0.3:netbios-ns   *:*
udp    0   0 *:netbios-ns        *:*
udp    0   0 192.168.0.3:netbios-dgm   *:*
udp    0   0 *:netbios-dgm        *:*
udp    0   0 *:tftp           *:*
udp    0   0 *:999            *:*
udp    0   0 *:sunrpc          *:*
udp    0   0 *:ipp            *:*
udp    0   0 *:1022           *:*
udp    0   0 *:638            *:*
Active UNIX domain sockets (only servers)
Proto RefCnt Flags    Type    State     I-Node Path
unix 2   [ ACC ]   STREAM   LISTENING   10621 @/tmp/fam-root-
unix 2   [ ACC ]   STREAM   LISTENING   7096  /var/run/acpid.socket
unix 2   [ ACC ]   STREAM   LISTENING   9792  /tmp/.gdm_socket
unix 2   [ ACC ]   STREAM   LISTENING   9927  /tmp/.X11-unix/X0
unix 2   [ ACC ]   STREAM   LISTENING   10489 /tmp/ssh-lbUnUf4552/agent.4552
unix 2   [ ACC ]   STREAM   LISTENING   10558 /tmp/ksocket-root/kdeinit__0
unix 2   [ ACC ]   STREAM   LISTENING   10560 /tmp/ksocket-root/kdeinit-:0
unix 2   [ ACC ]   STREAM   LISTENING   10570 /tmp/.ICE-unix/dcop4664-1270815442
unix 2   [ ACC ]   STREAM   LISTENING   10843 /tmp/.ICE-unix/4735
unix 2   [ ACC ]   STREAM   LISTENING   10591 /tmp/ksocket-root/klauncherah3arc.slave-soc
unix 2   [ ACC ]   STREAM   LISTENING   7763  /var/run/iiim/.iiimp-unix/9010
unix 2   [ ACC ]   STREAM   LISTENING   11047 /tmp/orbit-root/linc-1291-0-1e92c8082411
unix 2   [ ACC ]   STREAM   LISTENING   11053 /tmp/orbit-root/linc-128e-0-dc070659cbb3
unix 2   [ ACC ]   STREAM   LISTENING   8020  /var/run/dbus/system_bus_socket
unix 2   [ ACC ]   STREAM   LISTENING   58927 /tmp/mcop-root/vm-dev-2c28-4beba75f
unix 2   [ ACC ]   STREAM   LISTENING   7860  /tmp/.font-unix/fs7100
unix 2   [ ACC ]   STREAM   LISTENING   7658  /dev/gpmctl
unix 2   [ ACC ]   STREAM   LISTENING   10498 @/tmp/dbus-s2MLJGO5Ci
```

Linux dnsconf命令

Linux dnsconf命令用于设置DNS服务器组态。

dnsconf实际上为linuxconf的符号连接，提供图形截面的操作方式，供管理员管理DNS服务器。

语法

```
dnsconf [--deldomain<域>][--delsecondary<域>][--newdomain<域>][--set<主机><IP>][--setcname<
```

参数说明：

- --deldomain<域> 删除域。
- --delsecondary<域> 删除次级域。
- --newdomain<域> 新增域。
- --set<主机><IP> 新增主机记录。
- --setcname<CNAME><主机> 设置<CNAME>。
- --setmx<域><主机> 指定域的邮件主机。
- --setns<域><主机> 指定域的DNS服务器。
- --unset<主机> 删除DNS中某台主机的记录。

Linux mesg命令

Linux mesg命令用于设置终端机的写入权限。

将mesg设置y时，其他用户可利用write指令将信息直接显示在您的屏幕上。

语法

```
mesg [ny]
```

参数：

- n 不允许气筒用户将信息直接显示在你的屏幕上。
- y 允许气筒用户将信息直接显示在你的屏幕上。

实例

允许其他用户发信息到当前终端。

root 的终端

```
# mesg y //在这个终端设置允许发送消息
```

其他普通用户的终端：

```
$ write root pts/4
hello
hello
EOF //Ctrl+D 结束输入
```

root 的终端 终端显示

```
#
Message from root@w3cschool.cc (as hnlinux) on pts/5 at 14:48 ...
hello
EOF
```

Linux httpd命令

Linux httpd命令是Apache HTTP服务器程序。

httpd为Apache HTTP服务器程序。直接执行程序可启动服务器的服务。

语法

```
httpd [-hlLStvVX][-c<httpd指令>][-C<httpd指令>][-d<服务器根目录>][-D<设定文件参数>][-f<设定文件>
```

参数说明：

- -c<httpd指令> 在读取配置文件前，先执行选项中的指令。
- -C<httpd指令> 在读取配置文件后，再执行选项中的指令。
- -d<服务器根目录> 指定服务器的根目录。
- -D<设定文件参数> 指定要传入配置文件的参数。
- -f<设定文件> 指定配置文件。
- -h 显示帮助。
- -l 显示服务器编译时所包含的模块。
- -L 显示httpd指令的说明。
- -S 显示配置文件中的设定。
- -t 测试配置文件的语法是否正确。
- -v 显示版本信息。
- -V 显示版本信息以及建立环境。
- -X 以单一程序的方式来启动服务器。

实例

检查配置文件语法错误

```
# httpd -t
httpd: Could not determine the server's fully qualified domain name, using 127.0.0.1 for
Syntax OK
```

启动httpd

```
httpd
httpd: Could not determine the server's fully qualified domain name, using 127.0.0.1 for
```

显示编译模块

```
# httpd -l
Compiled in modules:
 core.c
 prefork.c
 http_core.c
 mod_so.c
```

显示配置文件

```
# httpd -L>1.log|tail -n 20 1.log
Maximum number of children alive at the same time
Allowed in *.conf only outside , or
ServerLimit (prefork.c)
Maximum value of MaxClients for this run of Apache
Allowed in *.conf only outside , or
KeepAliveTimeout (http_core.c)
Keep-Alive timeout duration (sec)
Allowed in *.conf only outside , or
MaxKeepAliveRequests (http_core.c)
Maximum number of Keep-Alive requests per connection, or 0 for infinite
Allowed in *.conf only outside , or
KeepAlive (http_core.c)
Whether persistent connections should be On or Off
Allowed in *.conf only outside , or
LoadModule (mod_so.c)
a module name and the name of a shared object file to load it from
Allowed in *.conf only outside , or
LoadFile (mod_so.c)
shared object file or library to load into the server at runtime
Allowed in *.conf only outside , or
```

Linux ifconfig命令

Linux ifconfig命令用于显示或设置网络设备。

ifconfig可设置网络设备的状态，或是显示目前的设置。

语法

```
ifconfig [网络设备][down up -allmulti -arp -promisc][add<地址>][del<地址>][<hw<网络设备类型><
```

参数说明：

- add<地址> 设置网络设备IPv6的IP地址。
- del<地址> 删除网络设备IPv6的IP地址。
- down 关闭指定的网络设备。
- <hw<网络设备类型><硬件地址> 设置网络设备的类型与硬件地址。
- io_addr<I/O地址> 设置网络设备的I/O地址。
- irq<IRQ地址> 设置网络设备的IRQ。
- media<网络媒介类型> 设置网络设备的媒介类型。
- mem_start<内存地址> 设置网络设备在主内存所占用的起始地址。
- metric<数目> 指定在计算数据包的转送次数时，所要加上的数目。
- mtu<字节> 设置网络设备的MTU。
- netmask<子网掩码> 设置网络设备的子网掩码。
- tunnel<地址> 建立IPv4与IPv6之间的隧道通信地址。
- up 启动指定的网络设备。
- -broadcast<地址> 将要送往指定地址的数据包当成广播数据包来处理。
- -pointopoint<地址> 与指定地址的网络设备建立直接连线，此模式具有保密功能。
- -promisc 关闭或启动指定网络设备的promiscuous模式。
- [IP地址] 指定网络设备的IP地址。
- [网络设备] 指定网络设备的名称。

实例

显示网络设备信息

```
# ifconfig
eth0   Link encap:Ethernet HWaddr 00:50:56:0A:0B:0C
     inet addr:192.168.0.3 Bcast:192.168.0.255 Mask:255.255.255.0
     inet6 addr: fe80::250:56ff:fe0a:b0c/64 Scope:Link
     UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
     RX packets:172220 errors:0 dropped:0 overruns:0 frame:0
     TX packets:132379 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:1000
     RX bytes:87101880 (83.0 MiB) TX bytes:41576123 (39.6 MiB)
     Interrupt:185 Base address:0x2024

lo    Link encap:Local Loopback
     inet addr:127.0.0.1 Mask:255.0.0.0
     inet6 addr: ::1/128 Scope:Host
     UP LOOPBACK RUNNING MTU:16436 Metric:1
     RX packets:2022 errors:0 dropped:0 overruns:0 frame:0
     TX packets:2022 errors:0 dropped:0 overruns:0 carrier:0
     collisions:0 txqueuelen:0
     RX bytes:2459063 (2.3 MiB) TX bytes:2459063 (2.3 MiB)
```

启动关闭指定网卡

```
# ifconfig eth0 down
# ifconfig eth0 up
```

为网卡配置和删除IPv6地址

```
# ifconfig eth0 add 33ffe:3240:800:1005::2/ 64 //为网卡诶之IPv6地址

# ifconfig eth0 del 33ffe:3240:800:1005::2/ 64 //为网卡删除IPv6地址
```

用ifconfig修改MAC地址

```
# ifconfig eth0 down //关闭网卡
# ifconfig eth0 hw ether 00:AA:BB:CC:DD:EE //修改MAC地址
# ifconfig eth0 up //启动网卡
# ifconfig eth1 hw ether 00:1D:1C:1D:1E //关闭网卡并修改MAC地址
# ifconfig eth1 up //启动网卡
```

配置IP地址

```
# ifconfig eth0 192.168.1.56
//给eth0网卡配置IP地址
# ifconfig eth0 192.168.1.56 netmask 255.255.255.0
// 给eth0网卡配置IP地址,并加上子掩码
# ifconfig eth0 192.168.1.56 netmask 255.255.255.0 broadcast 192.168.1.255
// 给eth0网卡配置IP地址,加上子掩码,加上个广播地址
```

启用和关闭ARP协议

```
# ifconfig eth0 arp  //开启
# ifconfig eth0 -arp  //关闭
```

设置最大传输单元

```
# ifconfig eth0 mtu 1500
//设置能通过的最大数据包大小为 1500 bytes
```

Linux minicom命令

Linux minicom命令用于调制解调器通信程序。

minicom是一个相当受欢迎的PPP拨号连线程序。

语法

```
minicom [-8lmMostz][-a<on或0ff>][-c<on或off>][-C<取文件>][-d<编号>][-p<模拟终端机>][-S<script
```

参数说明：

- -8 不要修改任何8位编码的字符。
- -a<on或0ff> 设置终端机属性。
- -c<on或off> 设置彩色模式。
- -C<取文件> 指定取文件，并在启动时开启取功能。
- -d<编号> 启动或直接拨号。
- -l 不会将所有的字符都转成ASCII码。
- -m 以Alt或Meta键作为指令键。
- -M 与-m参数类似。
- -o 不要初始化调制解调器。
- -p <模拟终端机> 使用模拟终端机。
- -s 开启程序设置画面。
- -S<script文件> 在启动时，执行指定的script文件。
- -t 设置终端机的类型。
- -z 在终端机上显示状态列。
- [配置文件] 指定minicom配置文件。

Linux traceroute命令

Linux traceroute命令用于显示数据包到主机间的路径。

traceroute指令让你追踪网络数据包的路由途径，预设数据包大小是40Bytes，用户可另行设置。

语法

```
traceroute [-dFlnrvx][-f<存活数值>][-g<网关>...][-i<网络界面>][-m<存活数值>][-p<通信端口>][-s<来
```

参数说明：

- -d 使用Socket层级的排错功能。
- -f<存活数值> 设置第一个检测数据包的存活数值TTL的大小。
- -F 设置勿离断位。
- -g<网关> 设置来源路由网关，最多可设置8个。
- -i<网络界面> 使用指定的网络界面送出数据包。
- -I 使用ICMP回应取代UDP资料信息。
- -m<存活数值> 设置检测数据包的最大存活数值TTL的大小。
- -n 直接使用IP地址而非主机名称。
- -p<通信端口> 设置UDP传输协议的通信端口。
- -r 忽略普通的Routing Table，直接将数据包送到远端主机上。
- -s<来源地址> 设置本地主机送出数据包的IP地址。
- -t<服务类型> 设置检测数据包的TOS数值。
- -v 详细显示指令的执行过程。
- -w<超时秒数> 设置等待远端主机回报的时间。
- -x 开启或关闭数据包的正确性检验。

实例

显示到达目的地的数据包路由

```
# traceroute www.google.com
traceroute: Warning: www.google.com has multiple addresses; using 66.249.89.99
traceroute to www.l.google.com (66.249.89.99), 30 hops max, 38 byte packets
1 192.168.0.1 (192.168.0.1) 0.653 ms 0.846 ms 0.200 ms
2 118.250.4.1 (118.250.4.1) 36.610 ms 58.438 ms 55.146 ms
3 222.247.28.177 (222.247.28.177) 54.809 ms 39.879 ms 19.186 ms
4 61.187.255.253 (61.187.255.253) 18.033 ms 49.699 ms 72.147 ms
5 61.137.2.177 (61.137.2.177) 32.912 ms 72.947 ms 41.809 ms
6 202.97.46.5 (202.97.46.5) 60.436 ms 25.527 ms 40.023 ms
7 202.97.35.69 (202.97.35.69) 40.049 ms 66.091 ms 44.358 ms
8 202.97.35.110 (202.97.35.110) 42.140 ms 70.913 ms 41.144 ms
9 202.97.35.14 (202.97.35.14) 116.929 ms 57.081 ms 60.336 ms
10 202.97.60.34 (202.97.60.34) 54.871 ms 69.302 ms 64.353 ms
11 * * *
12 209.85.255.80 (209.85.255.80) 95.954 ms 79.844 ms 76.052 ms
   MPLS Label=385825 CoS=5 TTL=1 S=0
13 209.85.249.195 (209.85.249.195) 118.687 ms 120.905 ms 113.936 ms
14 72.14.236.126 (72.14.236.126) 115.843 ms 137.109 ms 186.491 ms
15 nrt04s01-in-f99.1e100.net (66.249.89.99) 168.024 ms 140.551 ms 161.127 ms
```

Linux talk命令

Linux talk命令用于与其他使用者对谈。

使用权限：所有使用者。

语法

```
talk person [ttyname]
```

参数说明：

- person : 预备对谈的使用者帐号，如果该使用者在其他机器上，则可输入 person@machine.name
- ttyname : 如果使用者同时有两个以上的 tty 连线，可以自行选择合适的 tty 传讯息

实例

与现在机器上的使用者Rollaend对谈，此时 Rollaend 只有一个连线

```
talk Rollaend
```

接下来就是等Rollaend回应，若Rollaend接受，则Rollaend输入 `talk jzlee` 即可开始对谈，结束请按 ctrl+c

与linuxfab.cx上的使用者Rollaend对谈，使用pts/2来对谈

```
talk Rollaend@linuxfab.cx pts/2
```

接下来就是等Rollaend回应，若Rollaend接受，则Rollaend输入 `talk jzlee@jzlee.home` 即可开始对谈，结束请按 ctrl+c

注意：若萤幕的字会出现不正常的字元，试著按 ctrl+l 更新萤幕画面。

Linux ping命令

Linux ping命令用于检测主机。

执行ping指令会使用ICMP传输协议，发出要求回应的信息，若远端主机的网络功能没有问题，就会回应该信息，因而得知该主机运作正常。

语法

```
ping [-dfnqrRv][-c<完成次数>][-i<间隔秒数>][-I<网络界面>][-l<前置载入>][-p<范本样式>][-s<数据包大
```

参数说明：

- -d 使用Socket的SO_DEBUG功能。
- -c<完成次数> 设置完成要求回应的次数。
- -f 极限检测。
- -i<间隔秒数> 指定收发信息的间隔时间。
- -I<网络界面> 使用指定的网络界面送出数据包。
- -l<前置载入> 设置在送出要求信息之前，先行发出的数据包。
- -n 只输出数值。
- -p<范本样式> 设置填满数据包的范本样式。
- -q 不显示指令执行过程，开头和结尾的相关信息除外。
- -r 忽略普通的Routing Table，直接将数据包送到远端主机上。
- -R 记录路由过程。
- -s<数据包大小> 设置数据包的大小。
- -t<存活数值> 设置存活数值TTL的大小。
- -v 详细显示指令的执行过程。

实例

检测是否与主机连通

```
# ping www.w3cschool.cc //ping主机
PING aries.m.alikunlun.com (114.80.174.110) 56(84) bytes of data.
64 bytes from 114.80.174.110: icmp_seq=1 ttl=64 time=0.025 ms
64 bytes from 114.80.174.110: icmp_seq=2 ttl=64 time=0.036 ms
64 bytes from 114.80.174.110: icmp_seq=3 ttl=64 time=0.034 ms
64 bytes from 114.80.174.110: icmp_seq=4 ttl=64 time=0.034 ms
64 bytes from 114.80.174.110: icmp_seq=5 ttl=64 time=0.028 ms
64 bytes from 114.80.174.110: icmp_seq=6 ttl=64 time=0.028 ms
64 bytes from 114.80.174.110: icmp_seq=7 ttl=64 time=0.034 ms
64 bytes from 114.80.174.110: icmp_seq=8 ttl=64 time=0.034 ms
64 bytes from 114.80.174.110: icmp_seq=9 ttl=64 time=0.036 ms
64 bytes from 114.80.174.110: icmp_seq=10 ttl=64 time=0.041 ms

--- aries.m.alikunlun.com ping statistics ---
10 packets transmitted, 30 received, 0% packet loss, time 29246ms
rtt min/avg/max/mdev = 0.021/0.035/0.078/0.011 ms

//需要手动终止Ctrl+C
```

指定接收包的次数

```
# ping -c 2 www.w3cschool.cc
PING aries.m.alikunlun.com (114.80.174.120) 56(84) bytes of data.
64 bytes from 114.80.174.120: icmp_seq=1 ttl=54 time=6.18 ms
64 bytes from 114.80.174.120: icmp_seq=2 ttl=54 time=15.4 ms

--- aries.m.alikunlun.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1016ms
rtt min/avg/max/mdev = 6.185/10.824/15.464/4.640 ms

//收到两次包后，自动退出
```

多参数使用

```
# ping -i 3 -s 1024 -t 255 g.cn //ping主机
PING g.cn (203.208.37.104) 1024(1052) bytes of data.
1032 bytes from bg-in-f104.1e100.net (203.208.37.104): icmp_seq=0 ttl=243 time=62.5 ms
1032 bytes from bg-in-f104.1e100.net (203.208.37.104): icmp_seq=1 ttl=243 time=63.9 ms
1032 bytes from bg-in-f104.1e100.net (203.208.37.104): icmp_seq=2 ttl=243 time=61.9 ms

--- g.cn ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 6001ms
rtt min/avg/max/mdev = 61.959/62.843/63.984/0.894 ms, pipe 2
[root@linux ~]#

//-i 3 发送周期为 3秒 -s 设置发送包的大小 -t 设置TTL值为 255
```

Linux pppstats命令

Linux pppstats命令用于显示PPP连线状态。

利用pppstats(point to point protocol status)指令可让你得知PPP连接网络的相关信息。

语法

```
pppstats [-adrv][-c<执行次数>][-w<间隔秒数>][网络界面]
```

参数说明：

- -a 显示绝对统计值。
- -c<执行次数> 设置回报状况的次数。
- -d 显示相对统计值。
- -r 显示数据包压缩比率的统计值。
- -v 显示VJTCP文件头的压缩效率统计值。
- -w<间隔秒数> 设置显示统计信息的间隔时间。

实例

显示ppp的了连接状态

```
# pppstats
```

Linux samba命令

Linux samba命令用于Samba服务器控制。

samba为script文件，可启动，停止Samba服务器或回报目前的状态。

语法

```
samba [start][stop][status][restart]
```

参数说明：

- start 启动Samba服务器的服务。
- stop 停止Samba服务器的服务。
- status 显示Samba服务器目前的状态。
- restart 重新启动Samba服务器。

实例

启动Samba

```
# samba start
```

Linux statserial命令

Linux statserial命令用于显示串口状态。

statserial(status ofserial port)可显示各个接脚的状态，常用来判断串口是否正常。

语法

```
statserial [-dnx][串口设备名称]
```

参数说明：

- -d 以10进制数字来表示串口的状态。
- -n 仅显示一次串口的状态后即结束程序。
- -x 与-n参数类似，但是以16进制来表示。

实例

显示串口状态

```
# statserial /dev/tty1
```

只显示一次串口状态

```
# statserial -n /dev/tty1
```

Linux write命令

Linux write命令用于传讯息给其他使用者。

使用权限：所有使用者。

语法

```
write user [ttyname]
```

参数说明：

- user : 预备传讯息的使用者帐号
- ttyname : 如果使用者同时有两个以上的 tty 连线，可以自行选择合适的 tty 传讯息

实例

传讯息给 Rollaend，此时 Rollaend 只有一个连线

```
write Rollaend
```

接下来就是将讯息打上去，结束请按 ctrl+c

传讯息给 Rollaend，Rollaend 的连线有 pts/2，pts/3

```
write Rollaend pts/2
```

接下来就是将讯息打上去，结束请按 ctrl+c

注意：若对方设定 mesg n，则此时讯席将无法传给对方。

Linux setserial命令

Linux setserial命令用于设置或显示串口的相关信息。

setserial可用来设置串口或显示目前的设置。

语法

```
setserial [-abgGqvVz][设备][串口参数]
```

参数说明：

- -a 显示详细信息。
- -b 显示摘要信息。
- -g 显示串口的相关信息。
- -G 以指令列表的格式来显示信息。
- -q 执行时显示较少的信息。
- -v 执行时显示较多的信息。
- -V 显示版本信息。
- -z 设置前，先将所有的标记归零。

实例

显示串口信息

```
# setserial -g /dev/ttyS2
/dev/ttyS2, UART: unknown, Port: 0x03e8, IRQ: 4
```

Linux tty命令

Linux tty命令用于显示终端机连接标准输入设备的文件名称。

在Linux操作系统中，所有外围设备都有其名称与代号，这些名称代号以特殊文件的类型存放于/dev目录下。你可以执行tty(teletypewriter)指令查询目前使用的终端机的文件名称。

语法

```
tty [-s][--help][--version]
```

参数说明：

- -s或--silent或--quiet 不显示任何信息，只回传状态代码。
- --help 在线帮助。
- --version 显示版本信息。

实例

显示当前终端

```
# tty
/dev/pts/4
```

Linux newaliases命令

Linux newaliases命令会使用一个在 /etc/aliases 中的档案做使用者名称转换的动作。当 sendmail 收到一个要送给 xxx 的信时，它会依据 aliases档的内容送给另一个使用者。这个功能可以创造一个只有在信件系统内才有效的使用者。例如 mailing list 就会用到这个功能，在 mailinglist 中，我们可能会创造一个叫 redlinux@link.ece.uci.edu 的 mailinglist，但实际上并没有一个叫 redlinux 的使用者。实际 aliases 档的内容是将送给这个使用者的信都收给 mailing list 处理程序负责分送的工作。

/etc/aliases 是一个文字模式的档案，sendmail 需要一个二进位格式的 /etc/aliases.db。newaliases 的功能传是将 /etc/aliases 转换成一个 sendmail 所能了解的数据库。

使用权限：系统管理者。

语法

```
newaliases
```

参数说明：没有任何参数。

实例

```
# newaliases
```

下面命令会做相同的事

```
# sendmail -bi
```

Linux uuname命令

Linux uuname命令用于显示全部的UUCP远端主机。

uuname可显示UUCP远端主机。

语法

```
uuname [-alv][-I<配置文件>][--help]
```

参数说明：

- -a或--aliases 显示别名。
- -I<配置文件>或--config<配置文件> 指定程序的配置文件。
- -l或--local 显示本机名称。
- -v或--version 显示版本信息。
- --help 显示帮助。

实例

显示uucp主机名称

```
# uuname
```

Linux netconf命令

Linux netconf命令用于设置各项网络功能。

netconf是Red Hat Linux发行版专门用来调整Linux各项设置的程序。

语法

```
netconf
```

Linux smbd命令

Linux smbd命令用于Samba服务器程序。

smbd为Samba服务器程序，可分享文件与打印机等网络资源供Windows相关的用户端程序存取。

语法

```
smbd [-aDhoP][-d<排错层级>][-i<范围>][-l<记录文件>][-O<连接槽选项>][-p<连接端口编号>][-s<配置文件
```

参数说明：

- -a 所有的连线记录都会加到记录文件中。
- -d<排错层级> 指定记录文件所记载事件的详细程度。
- -D 使用此参数时，smbd会以服务程序的方式在后台执行。
- -h 显示帮助。
- -i<范围> 指定NetBIOS名称的范围。
- -l<记录文件> 指定记录文件的名称。
- -o 每次启动时，会覆盖原有的记录文件。
- -O<连接槽选项> 设置连接槽选项。
- -p<连接端口编号> 设置连接端口编号。
- -P 仅用来测试smbd程序的正确性。
- -s<配置文件> 指定smbd的设置文件。

实例

启动Samba服务器

```
# smbd -D
```

Linux ytalk命令

Linux ytalk命令用于与其他用户交谈。

通过ytalk指令，你可以和其他用户线上交谈，如果想和其他主机的用户交谈，在用户名称后加上其主机名称或IP地址即可。

语法

```
ytalk [-isxY][-h<主机名称IP地址>][用户名称...]
```

参数说明：

- -h<主机名称IP地址> 指定交谈对象所在的远端主机。
- -i 用提醒声响代替显示信息。
- -s 在指令提示符号先开启ytalk交谈窗。
- -x 关闭图形界面。
- -Y 所有必须回应yes或no的问题，都必须用大写英文字母"Y"或"N"回答。

实例

发送消息

```
# who  //显示当前用户
root   :0      Apr 9 20:17
root   pts/1    Apr 9 20:17
w3c   pts/6    May 27 16:47 (192.168.0.1)
root   pts/2    May 27 17:37 (192.168.0.1)
# ytalk w3c //发送消息
hey
```

Linux tcpdump命令

Linux tcpdump命令用于倾倒网络传输数据。

执行tcpdump指令可列出经过指定网络界面的数据包文件头，在Linux操作系统中，你必须是系统管理员。

语法

```
tcpdump [-adeflnNOpqStvx][-c<数据包数目>][-dd][-ddd][-F<表达文件>][-i<网络界面>][-r<数据包文件>
```

参数说明：

- -a 尝试将网络和广播地址转换成名称。
- -c<数据包数目> 收到指定的数据包数目后，就停止进行倾倒操作。
- -d 把编译过的数据包编码转换成可阅读的格式，并倾倒到标准输出。
- -dd 把编译过的数据包编码转换成C语言的格式，并倾倒到标准输出。
- -ddd 把编译过的数据包编码转换成十进制数字的格式，并倾倒到标准输出。
- -e 在每列倾倒资料上显示连接层级的文件头。
- -f 用数字显示网际网络地址。
- -F<表达文件> 指定内含表达方式的文件。
- -i<网络界面> 使用指定的网络截面送出数据包。
- -l 使用标准输出列的缓冲区。
- -n 不把主机的网络地址转换成名字。
- -N 不列出域名。
- -O 不将数据包编码最佳化。
- -p 不让网络界面进入混杂模式。
- -q 快速输出，仅列出少数的传输协议信息。
- -r<数据包文件> 从指定的文件读取数据包数据。
- -s<数据包大小> 设置每个数据包的大小。
- -S 用绝对而非相对数值列出TCP关联数。
- -t 在每列倾倒资料上不显示时间戳记。
- -tt 在每列倾倒资料上显示未经格式化的时间戳记。
- -T<数据包类型> 强制将表达方式所指定的数据包转译成设置的数据包类型。
- -v 详细显示指令执行过程。
- -vv 更详细显示指令执行过程。
- -x 用十六进制字码列出数据包资料。
- -w<数据包文件> 把数据包数据写入指定的文件。

实例

显示TCP包信息

```
# tcpdump
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
23:35:55.129998 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 148872068:148872168(100) ack 418
23:35:55.182357 IP 192.168.0.1.2101 > 192.168.0.3.ssh: . ack 100 win 64240
23:35:55.182397 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 100:200(100) ack 1 win 2100
23:35:55.131713 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: 50226+ PTR? 1.0.168.192.in-a
23:35:55.131896 PPPoE [ses 0x1cb0] IP 118.250.6.85.64215 > dns2.cs.hn.cn.domain: 50226+ P
23:35:55.154238 PPPoE [ses 0x1cb0] IP dns2.cs.hn.cn.domain > 118.250.6.85.64215: 50226 NX
23:35:55.156298 IP dns2.cs.hn.cn.domain > 192.168.0.3.32804: 50226 NXDomain 0/0/0 (42)
23:35:55.159292 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: 30304+ PTR? 3.0.168.192.in-a
23:35:55.159449 PPPoE [ses 0x1cb0] IP 118.250.6.85.64215 > dns2.cs.hn.cn.domain: 30304+ P
23:35:55.179816 PPPoE [ses 0x1cb0] IP dns2.cs.hn.cn.domain > 118.250.6.85.64215: 30304 NX
23:35:55.181279 IP dns2.cs.hn.cn.domain > 192.168.0.3.32804: 30304 NXDomain 0/0/0 (42)
23:35:55.181806 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 200:268(68) ack 1 win 2100
23:35:55.182177 IP 192.168.0.1.2101 > 192.168.0.3.ssh: . ack 268 win 64198
23:35:55.182677 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: 43983+ PTR? 112.96.103.202.i
23:35:55.182807 PPPoE [ses 0x1cb0] IP 118.250.6.85.64215 > dns2.cs.hn.cn.domain: 43983+ P
23:35:55.183055 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 268:352(84) ack 1 win 2100
23:35:55.201096 PPPoE [ses 0x1cb0] IP dns2.cs.hn.cn.domain > 118.250.6.85.64215: 43983 1/
23:35:55.203087 IP dns2.cs.hn.cn.domain > 192.168.0.3.32804: 43983 1/0/0 (72)
23:35:55.204666 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 352:452(100) ack 1 win 2100
23:35:55.204852 IP 192.168.0.1.2101 > 192.168.0.3.ssh: . ack 452 win 64152
23:35:55.205305 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 452:520(68) ack 1 win 2100
23:35:55.205889 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: 9318+ PTR? 85.6.250.118.in-a
23:35:55.206071 PPPoE [ses 0x1cb0] IP 118.250.6.85.64215 > dns2.cs.hn.cn.domain: 9318+ PT
23:35:55.215338 PPPoE [ses 0x1cb0] IP 115.238.1.45.3724 > 118.250.6.85.64120: P 239275192
23:35:55.216273 IP 115.238.1.45.3724 > 192.168.0.65.2057: P 2392751922:2392751987(65) ack
23:35:55.329204 IP 192.168.0.1.2101 > 192.168.0.3.ssh: . ack 520 win 64135
23:35:55.458214 IP 192.168.0.65.2057 > 115.238.1.45.3724: . ack 65 win 32590
23:35:55.458221 PPPoE [ses 0x1cb0] IP 118.250.6.85.64120 > 115.238.1.45.3724: . ack 65 wi
23:35:55.708228 PPPoE [ses 0x1cb0] IP 115.238.1.45.3724 > 118.250.6.85.64120: P 65:118(53
23:35:55.710213 IP 115.238.1.45.3724 > 192.168.0.65.2057: P 65:118(53) ack 1 win 54
23:35:55.865151 IP 192.168.0.65.2057 > 115.238.1.45.3724: . ack 118 win 32768
23:35:55.865157 PPPoE [ses 0x1cb0] IP 118.250.6.85.64120 > 115.238.1.45.3724: . ack 118 w
23:35:56.242805 IP 192.168.0.65.2057 > 115.238.1.45.3724: P 1:25(24) ack 118 win 32768
23:35:56.242812 PPPoE [ses 0x1cb0] IP 118.250.6.85.64120 > 115.238.1.45.3724: P 1:25(24)
23:35:56.276816 PPPoE [ses 0x1cb0] IP 115.238.1.45.3724 > 118.250.6.85.64120: . ack 25 wi
23:35:56.278240 IP 115.238.1.45.3724 > 192.168.0.65.2057: . ack 25 win 54
23:35:56.349747 PPPoE [ses 0x1cb0] IP 115.238.1.45.3724 > 118.250.6.85.64120: P 118:159(4
23:35:56.351780 IP 115.238.1.45.3724 > 192.168.0.65.2057: P 118:159(41) ack 25 win 54
23:35:56.400051 PPPoE [ses 0x1cb0] IP 119.147.18.44.8000 > 118.250.6.85.4000: UDP, length
23:35:56.475050 IP 192.168.0.65.2057 > 115.238.1.45.3724: . ack 159 win 32762
23:35:56.475063 PPPoE [ses 0x1cb0] IP 118.250.6.85.64120 > 115.238.1.45.3724: . ack 159 w
23:35:56.508968 PPPoE [ses 0x1cb0] IP 115.238.1.45.3724 > 118.250.6.85.64120: P 159:411(2
23:35:56.510182 IP 115.238.1.45.3724 > 192.168.0.65.2057: P 159:411(252) ack 25 win 54
23:35:56.592028 PPPoE [ses 0x1cb0] IP 117.136.2.43.38959 > 118.250.6.85.63283: UDP, lengt

44 packets captured
76 packets received by filter
0 packets dropped by kernel
```

显示指定数量包

```
# tcpdump -c 20
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
23:36:28.949538 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 148875984:148876020(36) ack 4184
23:36:28.994325 IP 192.168.0.1.2101 > 192.168.0.3.ssh: . ack 36 win 64020
23:36:28.994368 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 36:72(36) ack 1 win 2100
23:36:28.950779 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: 18242+ PTR? 1.0.168.192.in-a
23:36:28.950948 PPPoE [ses 0x1cb0] IP 118.250.6.85.64215 > dns2.cs.hn.cn.domain: 18242+ P
23:36:28.960105 PPPoE [ses 0x1cb0] IP 222.82.119.41.13594 > 118.250.6.85.63283: UDP, leng
23:36:28.962192 IP 222.82.119.41.13594 > 192.168.0.65.13965: UDP, length 36
23:36:28.963118 IP 192.168.0.65.13965 > 222.82.119.41.13594: UDP, length 34
23:36:28.963123 PPPoE [ses 0x1cb0] IP 118.250.6.85.63283 > 222.82.119.41.13594: UDP, leng
23:36:28.970185 PPPoE [ses 0x1cb0] IP dns2.cs.hn.cn.domain > 118.250.6.85.64215: 18242 NX
23:36:28.970413 IP dns2.cs.hn.cn.domain > 192.168.0.3.32804: 18242 NXDomain 0/0/0 (42)
23:36:28.972352 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: 17862+ PTR? 3.0.168.192.in-a
23:36:28.972474 PPPoE [ses 0x1cb0] IP 118.250.6.85.64215 > dns2.cs.hn.cn.domain: 17862+ P
23:36:28.982287 PPPoE [ses 0x1cb0] IP 121.12.131.163.13109 > 118.250.6.85.63283: UDP, len
23:36:28.984162 IP 121.12.131.163.13109 > 192.168.0.65.13965: UDP, length 27
23:36:28.985021 IP 192.168.0.65.13965 > 121.12.131.163.13109: UDP, length 103
23:36:28.985027 PPPoE [ses 0x1cb0] IP 118.250.6.85.63283 > 121.12.131.163.13109: UDP, len
23:36:28.991919 PPPoE [ses 0x1cb0] IP dns2.cs.hn.cn.domain > 118.250.6.85.64215: 17862 NX
23:36:28.993142 IP dns2.cs.hn.cn.domain > 192.168.0.3.32804: 17862 NXDomain 0/0/0 (42)
23:36:28.993574 IP 192.168.0.3.ssh > 192.168.0.1.2101: P 72:140(68) ack 1 win 2100
20 packets captured
206 packets received by filter
129 packets dropped by kernel
```

精简显示

```
# tcpdump -c 10 -q //精简模式显示 10个包
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
23:43:05.792280 IP 192.168.0.3.ssh > 192.168.0.1.2101: tcp 36
23:43:05.842115 IP 192.168.0.1.2101 > 192.168.0.3.ssh: tcp 0
23:43:05.845074 IP 115.238.1.45.3724 > 192.168.0.65.2057: tcp 0
23:43:05.907155 IP 192.168.0.3.ssh > 192.168.0.1.2101: tcp 36
23:43:05.793880 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: UDP, length 42
23:43:05.794076 PPPoE [ses 0x1cb0] IP 118.250.6.85.64219 > dns2.cs.hn.cn.domain: UDP, len
23:43:05.811127 PPPoE [ses 0x1cb0] IP dns2.cs.hn.cn.domain > 118.250.6.85.64219: UDP, len
23:43:05.814764 IP dns2.cs.hn.cn.domain > 192.168.0.3.32804: UDP, length 42
23:43:05.816404 IP 192.168.0.3.32804 > dns2.cs.hn.cn.domain: UDP, length 42
23:43:05.816545 PPPoE [ses 0x1cb0] IP 118.250.6.85.64219 > dns2.cs.hn.cn.domain: UDP, len
10 packets captured
39 packets received by filter
0 packets dropped by kernel
```

转换克阅读格式

```
# tcpdump -d
(000) ret   #96
```

转换成十进制格式

```
# tcpdump -ddd
1
6 0 0 96
```

Linux cu命令

Linux cu命令用于连接另一个系统主机。

cu(call up)指令可连接另一台主机，并采用类似拨号终端机的接口工作，也可执行简易的文件传输作业。

语法

```
cu [dehnotv][-a<通信端口>][-c<电话号码>][-E<脱离字符>][-I<设置文件>][-l<外围设备代号>][-s<连线速率
```

参数说明：

- -a<通信端口>或-p<通信端口>或--port<通信端口> 使用指定的通信端口进行连线。
- -c<电话号码>或--phone<电话号码> 拨打该电话号码。
- -d 进入排错模式。
- -e或--parity=even 使用双同位检查。
- -E<脱离字符>或--escape<脱离字符> 设置脱离字符。
- -h或--halfduple 使用半双工模式。
- -I<配置文件>或--config<配置文件> 指定要使用的配置文件。
- -l<外围设备代号>或--line<外围设备代号> 指定某项外围设备，作为连接的设备。
- -n或--prompt 拨号时等待用户输入电话号码。
- -o或--parity=odd 使用单同位检查。
- -s<连线速率>或--speed<连线速率>或--baud<连线速率>或-<连线速率> 设置连线的速率，单位以鲍率计算。
- -t或--maper 把CR字符置换成LF+CR字符。
- -v或--version 显示版本信息。
- -x<排错模式>或--debug<排错模式> 使用排错模式。
- -z<系统主机>或--system<系统主机> 连接该系统主机。
- --help 在线帮助。
- --nostop 关闭Xon/Xoff软件流量控制。
- --parity=none 不使用同位检查。

实例

与远程主机连接

```
# cu -c 0102377765
```

Linux efax命令

Linux efax命令用于收发传真。

支持Class 1与Class 2的调制解调器来收发传真。

语法

```
efax [-sw][-a<AT指令>][-c<调制解调器属性>][-d<驱动程序>][-f<字体文件>][-g<指令>][-h<传真标题字符串
```

参数说明：

- -a<AT指令> 以指定的AT指令来接电话。
- -c<调制解调器属性> 设置本机调制解调器的属性。
- -d<驱动程序> 指定调制解调器驱动程序。
- -f<字体文件> 使用指定的字体文件来建立传真标题。
- -g<指令> 若接到的电话为数据，则执行指定的指令。
- -h<传真标题字符串> 指定字符串为每页最前端的标题。
- -i<AT指令> 在调制解调器进入传真模式前，传送AT指令到调制解调器。
- -j<AT指令> 在调制解调器进入传真模式后，传送AT指令到调制解调器。
- -k<AT指令> 在调制解调器离开传真模式前，传送AT指令到调制解调器。
- -l<识别码> 设置本机调制解调器的识别码。
- -o<选项> 使用非标准调制解调器时设置相关选项。
- -q<错误次数> 接收传真时，当每页发生错误次数超过指定的数目时，要求对方重发。
- -r<文件名> 在接收传真时，将每页分别保存成文件。
- -v<信息类型> 选择要印出的信息类型。
- -w 不要接听电话，等待OK或CONNECT的信号。
- -x<UUCP锁定文件> 使用UUCP格式的锁定文件来锁定调制解调器。
- -t<电话号码><传真文件> 以<电话号码>中的号码来拨号，并将<传真文件>传真出去。

Linux pppsetup命令

Linux pppsetup命令用于设置PPP连线。

这是Slackware发行版内附程序，它具有互动式的问答界面，让用户轻易完成PPP的连线设置。

语法

```
pppsetup
```

实例

设置ppp拨号

```
# pppsetup
```

Linux testparm命令

Linux testparm命令用于测试Samba的设置是否正确无误。

执行testparm(test parameter)指令可以简单测试Samba的配置文件，假如测试结果无误，Samba常驻服务就能正确载入该设置值，但并不保证其后的操作如预期般一切正常。

语法

```
testparm [-s][配置文件][<主机名称><IP地址>]
```

参数说明：

- -s 不显示提示符号等待用户按下Enter键，就直接列出Samba服务定义信息。

实例

查看Ssmba配置

```
# testparm
Load smb config files from /etc/samba/smb.conf
Processing section '[homes]'
Processing section '[printers]'
Processing section '[uptech]'
Processing section '[home]'
Loaded services file OK.
Server role: ROLE_STANDALONE
Press enter to see a dump of your service definitions
      ///按下回车继续
# Global parameters
[global]
workgroup = MYGROUP
server string = Samba Server
security = SHARE
encrypt passwords = No
password server = None
log file = /var/log/samba/%m.log
max log size = 50
socket options = TCP_NODELAY SO_RCVBUF=8192 SO_SNDBUF=8192
printcap name = /etc/printcap
dns proxy = No
idmap uid = 16777216-33554431
idmap gid = 16777216-33554431
cups options = raw

[homes]
comment = Home Directories
read only = No
browseable = No

[printers]
comment = All Printers
path = /var/spool/samba
printable = Yes
browseable = No

[uptech]
comment = *
path = /home/uptech
read only = No
guest ok = Yes

[home]
comment = *
path = /home
read only = No
guest ok = Yes
```

Linux smbclient命令

Linux smbclient命令可存取SMB/CIFS服务器的用户端程序。

SMB与CIFS为服务器通信协议，常用于Windows95/98/NT等系统。smbclient(samba client)可让Linux系统存取Windows系统所分享的资源。

语法

```
smbclient [网络资源][密码][-EhLN][-B<IP地址>][-d<排错层级>][-i<范围>][-I<IP地址>][-l<记录文件>][
```

参数说明：

- [网络资源] [网络资源]的格式为//服务器名称/资源分享名称。
- [密码] 输入存取网络资源所需的密码。
- -B<IP地址> 传送广播数据包时所用的IP地址。
- -d<排错层级> 指定记录文件所记载事件的详细程度。
- -E 将信息送到标准错误输出设备。
- -h 显示帮助。
- -i<范围> 设置NetBIOS名称范围。
- -I<IP地址> 指定服务器的IP地址。
- -l<记录文件> 指定记录文件的名称。
- -L 显示服务器端所分享出来的所有资源。
- -M<NetBIOS名称> 可利用WinPopup协议，将信息送给选项中所指定的主机。
- -n<NetBIOS名称> 指定用户端所要使用的NetBIOS名称。
- -N 不用询问密码。
- -O<连接槽选项> 设置用户端TCP连接槽的选项。
- -p<TCP连接端口> 指定服务器端TCP连接端口编号。
- -R<名称解析顺序> 设置NetBIOS名称解析的顺序。
- -s<目录> 指定smb.conf所在的目录。
- -t<服务器字码> 设置用何种字符码来解析服务器端的文件名称。
- -T<tar选项> 备份服务器端分享的全部文件，并打包成tar格式的文件。
- -U<用户名称> 指定用户名称。
- -W<工作群组> 指定工作群组名称。

Linux shapecfg命令

Linux shapecfg命令用于管制网络设备的流量。

自Linux-2.15开始，便支持流量管制的功能。

语法

```
shapecfg attach [流量管制器][网络设备]
```

或

```
shapecfg speed [流量管制器][带宽]
```

参数说明：

- attach 将流量管制器与实际的网络设备结合。
- speed 设置流量管制器的对外传输带宽。

Linux命令大全 - 系统管理

adduser	chfn	useradd	date
exit	finger	fwhios	sleep
suspend	groupdel	groupmod	halt
kill	last	lastb	login
logname	logout	ps	nice
procinfo	top	pstree	reboot
rlogin	rsh	sliplogin	screen
shutdown	rwho	sudo	gitps
swatch	tload	logrotate	uname
chsh	userconf	userdel	usermod
vlock	who	whoami	whois
newgrp	renice	su	skill
w	id	free	

Linux date命令

Linux date命令可以用来显示或设定系统的日期与时间，在显示方面，使用者可以设定欲显示的格式，格式设定为一个加号后接数个标记，其中可用的标记列表如下：

时间方面：

- % : 印出 %
- %n : 下一行
- %t : 跳格
- %H : 小时(00..23)
- %I : 小时(01..12)
- %k : 小时(0..23)
- %l : 小时(1..12)
- %M : 分钟(00..59)
- %p : 显示本地 AM 或 PM
- %r : 直接显示时间 (12 小时制，格式为 hh:mm:ss [AP]M)
- %s : 从 1970 年 1 月 1 日 00:00:00 UTC 到目前为止的秒数
- %S : 秒(00..61)
- %T : 直接显示时间 (24 小时制)
- %X : 相当于 %H:%M:%S
- %Z : 显示时区

日期方面：

- %a : 星期几 (Sun..Sat)
- %A : 星期几 (Sunday..Saturday)
- %b : 月份 (Jan..Dec)
- %B : 月份 (January..December)
- %c : 直接显示日期与时间
- %d : 日 (01..31)
- %D : 直接显示日期 (mm/dd/yy)
- %h : 同 %b
- %j : 一年中的第几天 (001..366)
- %m : 月份 (01..12)
- %U : 一年中的第几周 (00..53) (以 Sunday 为一周的第一天的情形)
- %w : 一周中的第几天 (0..6)
- %W : 一年中的第几周 (00..53) (以 Monday 为一周的第一天的情形)
- %x : 直接显示日期 (mm/dd/yy)
- %y : 年份的最后两位数字 (00.99)

- %Y : 完整年份 (0000..9999)

若是不以加号作为开头，则表示要设定时间，而时间格式为 MMDDhhmm[[CC]YY][.ss]，其中 MM 为月份，DD 为日，hh 为小时，mm 为分钟，CC 为年份前两位数字，YY 为年份后两位数字，ss 为秒数。

使用权限：所有使用者。

当您不希望出现无意义的 0 时(比如说 1999/03/07)，则可以在标记中插入 - 符号，比如说 date '+%-H:%-M:%-S' 会把时分秒中无意义的 0 给去掉，像是原本的 08:09:04 会变为 8:9:4。另外，只有取得权限者(比如说 root)才能设定系统时间。

当您以 root 身分更改了系统时间之后，请记得以 clock -w 来将系统时间写入 CMOS 中，这样下次重新开机时系统时间才会持续抱持最新的正确值。

语法

```
date [-u] [-d datestr] [-s datestr] [--utc] [--universal] [--date=datestr] [--set=datestr
```

参数说明：

- -d datestr : 显示 datestr 中所设定的时间 (非系统时间)
- --help : 显示辅助讯息
- -s datestr : 将系统时间设为 datestr 中所设定的时间
- -u : 显示目前的格林威治时间
- --version : 显示版本编号

实例

显示当前时间

```
# date
三 5月 12 14:08:12 CST 2010
# date '+%c'
2010年05月12日 星期三 14时09分02秒
# date '+%D' //显示完整的时间
05/12/10
# date '+%x' //显示数字日期，年份两位数表示
2010年05月12日
# date '+%T' //显示日期，年份用四位数表示
14:09:31
# date '+%X' //显示24小时的格式
14时09分39秒
```

按自己的格式输出

```
# date '+usr_time: $1:%M %P -hey'
usr_time: $1:16 下午 -hey
```

显示时间后跳行，再显示目前日期

```
date '+%T%n%D'
```

显示月份与日数

```
date '+%B %d'
```

显示日期与设定时间(12:34:56)

```
date --date '12:34:56'
```

Linux chfn命令

Linux chfn命令提供使用者更改个人资讯，用于 finger and mail username

使用权限：所有使用者。

语法

```
shell>> chfn
```

实例

改变finger信息

```
# chfn
Changing finger information for root.
Name [root]: hnlinux
Office []: hn
Office Phone []: 888888
Home Phone []: 9999999

Finger information changed.
```

改变账号真实姓名

```
# chfn -f hnunix
Changing finger information for root.
Finger information changed.
```

Linux adduser命令

Linux adduser命令用于新增使用者帐号或更新预设的使用者资料。

adduser 与 useradd 指令为同一指令（经由符号连结 symbolic link）。

使用权限：系统管理员。

adduser是增加使用者。相对的，也有删除使用者的指令，userdel。语法:userdel [login ID]

语法

```
adduser [-c comment] [-d home_dir] [-e expire_date] [-f inactive_time] [-g initial_group]
```

或

```
adduser -D [-g default_group] [-b default_home] [-f default_inactive] [-e default_expire_
```

参数说明：

- -c comment 新使用者位于密码档（通常是 /etc/passwd）的注解资料
- -d home_dir 设定使用者的家目录为 home_dir，预设值为预设的 home 后面加上使用者帐号 loginid
- -e expire_date 设定此帐号的使用期限（格式为 YYYY-MM-DD），预设值为永久有效
- -f inactive_time 范例：

实例

添加一个一般用户

```
# useradd kk //添加用户kk
```

为添加的用户指定相应的用户组

```
# useradd ?g root kk //添加用户kk，并指定用户所在的组为root用户组
```

创建一个系统用户

```
# useradd ?r kk //创建一个系统用户kk
```

为新添加的用户指定/home目录

```
# useradd-d /home/myf kk //新添加用户kk，其home目录为/home/myf
//当用户名kk登录主机时，系统进入的默认目录为/home/myf
```

Linux groupdel命令

Linux groupdel命令用于删除群组。

需要从系统上删除群组时，可用groupdel(group delete)指令来完成这项工作。倘若该群组中仍包括某些用户，则必须先删除这些用户后，方能删除群组。

语法

```
groupdel [群组名称]
```

实例

删除一个群组

```
# groupdel hnuser
```

Linux useradd命令

Linux useradd命令用于建立用户帐号。

useradd可用来建立用户帐号。帐号建好之后，再用passwd设定帐号的密码．而可用userdel删除帐号。使用useradd指令所建立的帐号，实际上是保存在/etc/passwd文本文件中。

语法

```
useradd [-mMnr][-c <备注>][-d <登入目录>][-e <有效期限>][-f <缓冲天数>][-g <群组>][-G <群组>][-
```

或

```
useradd -D [-b][-e <有效期限>][-f <缓冲天数>][-g <群组>][-G <群组>][-s <shell>]
```

参数说明：

- -c<备注>　加上备注文字。备注文字会保存在passwd的备注栏位中。
- -d<登入目录>　指定用户登入时的启始目录。
- -D　变更预设值．
- -e<有效期限>　指定帐号的有效期限。
- -f<缓冲天数>　指定在密码过期后多少天即关闭该帐号。
- -g<群组>　指定用户所属的群组。
- -G<群组>　指定用户所属的附加群组。
- -m　自动建立用户的登入目录。
- -M　不要自动建立用户的登入目录。
- -n　取消建立以用户名称为名的群组．
- -r　建立系统帐号。
- -s<shell>　 指定用户登入后所使用的shell。
- -u<uid>　指定用户ID。

实例

添加一般用户

```
# useradd tt
```

为添加的用户指定相应的用户组

```
# useradd -g root tt
```

创建一个系统用户

```
# useradd -r tt
```

为新添加的用户指定home目录

```
# useradd -d /home/myd tt
```

建立用户且制定ID

```
# useradd caojh -u 544
```

Linux groupmod命令

Linux groupmod命令用于更改群组识别码或名称。

需要更改群组的识别码或名称时，可用groupmod指令来完成这项工作。

语法

```
groupmod [-g <群组识别码> <-o>][-n <新群组名称>][群组名称]
```

参数：

- -g <群组识别码>　设置欲使用的群组识别码。
- -o　重复使用群组识别码。
- -n <新群组名称>　设置欲使用的群组名称。

实例

修改组名

```
[root@w3cschool.cc ~]# groupadd linuxso
[root@w3cschool.cc ~]# tail -1 /etc/group
linuxso:x:500:
[root@w3cschool.cc ~]# tail -1 /etc/group
linuxso:x:500:
[root@w3cschool.cc ~]# groupmod -n linux linuxso
[root@w3cschool.cc ~]# tail -1 /etc/group
linux:x:500:
```

Linux logname命令

Linux logname命令用于显示用户名称。

执行logname指令，它会显示目前用户的名称。

语法

```
logname [--help][--version]
```

参数：

- --help 　在线帮助。
- --vesion 　显示版本信息。

实例

显示登录账号的信息：

```
# logname
root
```

Linux logout命令

Linux logout命令用于退出系统。

logout指令让用户退出系统，其功能和login指令相互对应。

语法

```
logout
```

实例

退出系统：

```
[root@w3cschool.cc ~]# logout
```

Linux ps命令

Linux ps命令用于显示当前进程 (process) 的状态。

语法

```
ps [options] [--help]
```

参数：

- ps 的参数非常多, 在此仅列出几个常用的参数并大略介绍含义
- -A 列出所有的行程
- -w 显示加宽可以显示较多的资讯
- -au 显示较详细的资讯
- -aux 显示所有包含其他使用者的行程
- au(x) 输出格式 :
- USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
- USER: 行程拥有者
- PID: pid
- %CPU: 占用的 CPU 使用率
- %MEM: 占用的记忆体使用率
- VSZ: 占用的虚拟记忆体大小
- RSS: 占用的记忆体大小
- TTY: 终端的次要装置号码 (minor device number of tty)
- STAT: 该行程的状态:
- D: 不可中断的静止 (通悸□□缜b进行 I/O 动作)
- R: 正在执行中
- S: 静止状态
- T: 暂停执行
- Z: 不存在但暂时无法消除
- W: 没有足够的记忆体分页可分配
- <: 高优先序的行程
- N: 低优先序的行程
- L: 有记忆体分页分配并锁在记忆体内 (实时系统或捱A I/O)
- START: 行程开始时间
- TIME: 执行的时间
- COMMAND:所执行的指令

实例

```
# ps -A 显示进程信息
PID TTY      TIME CMD
  1 ?    00:00:02 init
  2 ?    00:00:00 kthreadd
  3 ?    00:00:00 migration/0
  4 ?    00:00:00 ksoftirqd/0
  5 ?    00:00:00 watchdog/0
  6 ?    00:00:00 events/0
  7 ?    00:00:00 cpuset
  8 ?    00:00:00 khelper
  9 ?    00:00:00 netns
  10 ?    00:00:00 async/mgr
  11 ?    00:00:00 pm
  12 ?    00:00:00 sync_supers
  13 ?    00:00:00 bdi-default
  14 ?    00:00:00 kintegrityd/0
  15 ?    00:00:02 kblockd/0
  16 ?    00:00:00 kacpid
  17 ?    00:00:00 kacpi_notify
  18 ?    00:00:00 kacpi_hotplug
  19 ?    00:00:27 ata/0
……省略部分结果
30749 pts/0  00:00:15 gedit
30886 ?    00:01:10 qtcreator.bin
30894 ?    00:00:00 qtcreator.bin
31160 ?    00:00:00 dhclient
31211 ?    00:00:00 aptd
31302 ?    00:00:00 sshd
31374 pts/2  00:00:00 bash
31396 pts/2  00:00:00 ps
```

显示指定用户信息

```
# ps -u root //显示root进程用户信息
 PID TTY      TIME CMD
  1 ?    00:00:02 init
  2 ?    00:00:00 kthreadd
  3 ?    00:00:00 migration/0
  4 ?    00:00:00 ksoftirqd/0
  5 ?    00:00:00 watchdog/0
  6 ?    00:00:00 events/0
  7 ?    00:00:00 cpuset
  8 ?    00:00:00 khelper
  9 ?    00:00:00 netns
  10 ?    00:00:00 async/mgr
  11 ?    00:00:00 pm
  12 ?    00:00:00 sync_supers
  13 ?    00:00:00 bdi-default
  14 ?    00:00:00 kintegrityd/0
  15 ?    00:00:02 kblockd/0
  16 ?    00:00:00 kacpid
……省略部分结果
30487 ?    00:00:06 gnome-terminal
30488 ?    00:00:00 gnome-pty-helpe
30489 pts/0  00:00:00 bash
30670 ?    00:00:00 debconf-communi
30749 pts/0  00:00:15 gedit
30886 ?    00:01:10 qtcreator.bin
30894 ?    00:00:00 qtcreator.bin
31160 ?    00:00:00 dhclient
31211 ?    00:00:00 aptd
31302 ?    00:00:00 sshd
31374 pts/2  00:00:00 bash
31397 pts/2  00:00:00 ps
```

显示所有进程信息，连同命令行

```
# ps -ef //显示所有命令，连带命令行
UID    PID PPID C STIME TTY     TIME CMD
root     1   0 0 10:22 ?    00:00:02 /sbin/init
root     2   0 0 10:22 ?    00:00:00 [kthreadd]
root     3   2 0 10:22 ?    00:00:00 [migration/0]
root     4   2 0 10:22 ?    00:00:00 [ksoftirqd/0]
root     5   2 0 10:22 ?    00:00:00 [watchdog/0]
root     6   2 0 10:22 ?    /usr/lib/NetworkManager
……省略部分结果
root   31302 2095 0 17:42 ?    00:00:00 sshd: root@pts/2
root   31374 31302 0 17:42 pts/2  00:00:00 -bash
root   31400   1 0 17:46 ?    00:00:00 /usr/bin/python /usr/sbin/aptd
root   31407 31374 0 17:48 pts/2  00:00:00 ps -ef
```

Linux exit命令

Linux exit命令用于退出目前的shell。

执行exit可使shell以指定的状态值退出。若不设置状态值参数，则shell以预设值退出。状态值0代表执行成功，其他值代表执行失败。exit也可用在script，离开正在执行的script，回到shell。

语法

```
exit [状态值]
```

实例

退出终端

```
# exit
```

Linux finger命令

Linux finger命令可以让使用者查询一些其他使用者的资料。会列出来的资料有：

- Login Name
- User Name
- Home directory
- Shell
- Login status
- mail status
- .plan
- .project
- .forward

其中 .plan、.project 和 .forward 就是使用者在他的 Home Directory 里的 .plan， .project 和 .forward 等档案里的资料。如果没有就没有。finger 指令并不限定于在同一服务器上查询，也可以寻找某一个远端服务器上的使用者。只要给一个像是 E-mail address 一般的地址即可。

使用权限：所有使用者。

语法

```
finger [options] user[@address]
```

参数说明：

- -l 多行显示。
- -s 单行显示。这个选项只显示登入名称、真实姓名、终端机名称、闲置时间、登入时间、办公室号码及电话号码。如果所查询的使用者是远端服务器的使用者，这个选项无效。

实例

列出当前登录用户的相关信息

```
# finger -l //显示用户信息
Login: root Name: root
Directory: /root Shell: /bin/bash
On since Fri Apr 9 20:17 (CST) on :0 (messages off)
On since Fri Apr 9 20:17 (CST) on pts/1 32 days 22 hours idle
On since Fri Apr 9 20:17 (CST) on pts/3 4 hours 5 minutes idle
(messages off)
On since Wed May 12 18:08 (CST) on pts/4 from 192.168.1.10
On since Wed May 12 18:35 (CST) on pts/5 from 192.168.1.10
7 minutes 54 seconds idle
On since Wed May 12 14:37 (CST) on pts/2 from 192.168.1.10
3 hours 14 minutes idle
On since Wed May 12 14:53 (CST) on pts/7 34 minutes 25 seconds idle
(messages off)
On since Wed May 12 16:53 (CST) on pts/8 from 192.168.1.10
30 minutes 18 seconds idle
Mail last read Mon Mar 31 04:02 2008 (CST)
No Plan.
```

显示指定用户信息

```
# finger -m hnlinux
```

显示远程用户信息

```
# finger -m root@192.168.1.13
```

下列指令可以查询本机管理员的资料：

```
finger root
```

其结果如下：

```
Login: root Name: root
Directory: /root Shell: /bin/bash
Never logged in.
No mail.
No Plan.
```

Linux fwhios命令

Linux fwhios命令用于查找并显示用户信息。

本指令的功能有点类似finger指令，它会去查找并显示指定帐号的用户相关信息。不同之处在于fwhois指令是到Network Solutions的WHOIS数据库去查找，该帐号名称必须有在上面注册才能寻获，且名称没有大小写的差别。

语法

```
fwhios [帐号名称]
```

Linux sleep命令

Linux sleep命令可以用来将目前动作延迟一段时间。

使用权限：所有使用者。

语法

```
sleep [--help] [--version] number[smhd]
```

参数说明：

- --help : 显示辅助讯息
- --version : 显示版本编号
- number : 时间长度，后面可接 s、m、h 或 d
- 其中 s 为秒，m 为 分钟，h 为小时，d 为日数

实例

休眠5分钟

```
# sleep 5m
```

显示目前时间后延迟 1 分钟，之后再次显示时间

```
date;sleep 1m;date
```

Linux suspend命令

Linux suspend命令用于暂停执行shell。

suspend为shell内建指令，可暂停目前正在执行的shell。若要恢复，则必须使用SIGCONT信息。

语法

```
suspend [-f]
```

参数说明：

- -f 　若目前执行的shell为登入的shell，则suspend预设无法暂停此shell。若要强迫暂停登入的shell，则必须使用-f参数。

实例

暂停shell

```
# suspend
-bash: suspend: 无法挂起一个登录 shell
# suspend -f
```

Linux login命令

Linux login命令用于登入系统。

login指令让用户登入系统，您亦可通过它的功能随时更换登入身份。在Slackware发行版中，您可在指令后面附加欲登入的用户名称，它会直接询问密码，等待用户输入。当/etc目录里含名称为nologin的文件时，系统只root帐号登入系统，其他用户一律不准登入。

语法

```
login
```

实例

使用新的身份登录系统

```
# login
```

Linux lastb命令

Linux lastb命令用于列出登入系统失败的用户相关信息。

单独执行lastb指令，它会读取位于/var/log目录下，名称为btmp的文件，并把该文件内容记录的登入失败的用户名单，全部显示出来。

语法

```
lastb [-adRx][-f <记录文件>][-n <显示列数>][帐号名称...][终端机编号...]
```

参数说明：

- -a　把从何处登入系统的主机名称或IP地址显示在最后一行。
- -d　将IP地址转换成主机名称。
- -f<记录文件>　指定记录文件。
- -n<显示列数>或-<显示列数>　设置列出名单的显示列数。
- -R　不显示登入系统的主机名称或IP地址。
- -x　显示系统关机，重新开机，以及执行等级的改变等信息。

实例

显示登录失败的用户

```
# lastb
root    tty7      :1          Thu May 13 11:26 - 11:26 (00:00)

btmp begins Thu May 13 11:26:39 2014
```

Linux rlogin命令

Linux rlogin命令用于远端登入。

执行rlogin指令开启终端机阶段操作，并登入远端主机。

语法

```
rlogin [-8EL][-e <脱离字符>][-l <用户名称>][主机名称或IP地址]
```

必要参数：

- -E 忽略escape字符
- -8 只识别8位字的字符
- -L 允许rlogin会话运行在litout模式
- -ec 设置escape字符为c
- -c 断开连接前要求确认
- -a 强制要求远程主机在发送完一个空的本地用户名之后请求一个密码
- -f 向远端主机发送一个本地认证
- -F 向远程主机发送一个可转寄的本地认证
- -7 强制执行7为的传输
- -d 打开用于远端主机通信的TCP套接口的调试
- -k 要求包含远端主机的tisckets
- -x 启动数据传输的DES加密
- -4 只使用 kerkberos的版本4的认证

选择参数：

- -e<字符> 设置退出字符 -l<用户> 指定登陆的用户 -t<终端类型> 设置终端类型

实例

显示rlogin服务是否开启

```
# chkconfig --list //检测rlogin服务是否开启
```

开启rlogin服务

```
# chkconfig rlogin on //开启rlogin服务
```

登陆远程主机

```
# rlogin 192.168.1.88
Password：
Password：
Login incorrect
Login:root
Passwd:
Login incorrect
Login:kk
Passwd:
```

指定用户名登陆远程主机

```
# rlogin 192.168.1.88 -l hnlinux

Passord:
Last login：Mon May 28 15：30:25 from 192.168.1.88

#
```

Linux last命令

Linux last命令用于显示系统开机以来获是从每月初登入者的讯息。

使用权限：所有使用者。

语法

```
shell>> last [options]
```

参数说明：

- -R 省略 hostname 的栏位
- -num 展示前 num 个
- username 展示 username 的登入讯息
- tty 限制登入讯息包含终端机代号

实例

```
shell>> last -R -2
johnney pts/1 Mon Aug 14 20:42 still logged in
johnney pts/0 Mon Aug 14 19:59 still logged in
wtmp begins Tue Aug 1 09:01:10 2000 ### /var/log/wtmp
shell>> last -2 minery
minery pts/0 140.119.217.115 Mon Aug 14 18:37 - 18:40 (00:03)
minery pts/0 140.119.217.115 Mon Aug 14 17:22 - 17:24 (00:02)
wtmp begins Tue Aug 1 09:01:10 2000
```

一般显示方法

```
# last
```

简略显示，并指定显示的个数

```
# last -n 5 -R
root    pts/4    Thu May 13 17:25  still logged in
root    pts/2    Thu May 13 17:23 - 17:25 (00:02)
root    pts/1    Thu May 13 16:46  still logged in
root    pts/7    Thu May 13 15:36  still logged in
root    pts/9    Thu May 13 15:35  still logged in

wtmp begins Thu May 13 18:55:40 2014
```

显示最后一列显示主机IP地址

```
# last -n 5 -a -i
root   pts/4    Thu May 13 17:25  still logged in  192.168.1.10
root   pts/2    Thu May 13 17:23 - 17:25 (00:02)   192.168.1.10
root   pts/1    Thu May 13 16:46  still logged in  192.168.1.10
root   pts/7    Thu May 13 15:36  still logged in  192.168.1.10
root   pts/9    Thu May 13 15:35  still logged in  192.168.1.10

wtmp begins Thu May 13 18:55:40 2014
```

Linux reboot命令

Linux reboot命令用于用来重新启动计算机。

若系统的 runlevel 为 0 或 6 ，则重新开机，否则以 shutdown 指令（加上 -r 参数）来取代

语法

```
reboot [-n] [-w] [-d] [-f] [-i]
```

参数：

- -n : 在重开机前不做将记忆体资料写回硬盘的动作
- -w : 并不会真的重开机，只是把记录写到 /var/log/wtmp 档案里
- -d : 不把记录写到 /var/log/wtmp 档案里（-n 这个参数包含了 -d）
- -f : 强迫重开机，不呼叫 shutdown 这个指令
- -i : 在重开机之前先把所有网络相关的装置先停止

实例

重新启动

```
# reboot
```

Linux kill命令

Linux kill命令用于删除执行中的程序或工作。

kill可将指定的信息送至程序。预设的信息为SIGTERM(15)，可将指定程序终止。若仍无法终止该程序，可使用SIGKILL(9)信息尝试强制删除程序。程序或工作的编号可利用ps指令或jobs指令查看。

语法

```
kill [-s <信息名称或编号>][程序]  或  kill [-l <信息编号>]
```

参数说明：

- -l <信息编号>　若不加<信息编号>选项，则-l参数会列出全部的信息名称。
- -s <信息名称或编号>　指定要送出的信息。
- [程序]　[程序]可以是程序的PID或是PGID，也可以是工作编号。

实例

杀死进程

```
# kill 12345
```

强制杀死进程

```
# kill -KILL 123456
```

发送SIGHUP信号，可以使用一下信号

```
# kill -HUP pid
```

彻底杀死进程

```
# kill -9 123456
```

显示信号

```
# kill -l
1) SIGHUP      2) SIGINT      3) SIGQUIT      4) SIGILL      5) SIGTRAP
6) SIGABRT      7) SIGBUS      8) SIGFPE      9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT     17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG     24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM     27) SIGPROF     28) SIGWINCH     29) SIGIO     30) SIGPWR
31) SIGSYS     34) SIGRTMIN     35) SIGRTMIN+1     36) SIGRTMIN+2     37) SIGRTMIN+3
38) SIGRTMIN+4     39) SIGRTMIN+5     40) SIGRTMIN+6     41) SIGRTMIN+7     42) SIGRTMIN+8
43) SIGRTMIN+9     44) SIGRTMIN+10     45) SIGRTMIN+11     46) SIGRTMIN+12     47) SIGRTMIN+1
48) SIGRTMIN+14     49) SIGRTMIN+15     50) SIGRTMAX-14     51) SIGRTMAX-13     52) SIGRTMAX-
53) SIGRTMAX-11     54) SIGRTMAX-10     55) SIGRTMAX-9     56) SIGRTMAX-8     57) SIGRTMAX-7
58) SIGRTMAX-6     59) SIGRTMAX-5     60) SIGRTMAX-4     61) SIGRTMAX-3     62) SIGRTMAX-2
63) SIGRTMAX-1     64) SIGRTMAX
```

杀死指定用户所有进程

```
#kill -9 $(ps -ef | grep hnlinux) //方法一 过滤出hnlinux用户进程
#kill -u hnlinux //方法二
```

Linux halt命令

若系统的 runlevel 为 0 或 6 ，则Linux halt命令关闭系统，否则以 shutdown 指令（加上 -h 参数）来取代。

使用权限：系统管理者。

语法

```
halt [-n] [-w] [-d] [-f] [-i] [-p]
```

参数说明：

- -n : 在关机前不做将记忆体资料写回硬盘的动作
- -w : 并不会真的关机，只是把记录写到 /var/log/wtmp 档案里
- -d : 不把记录写到 /var/log/wtmp 档案里（-n 这个参数包含了 -d） -f : 强迫关机，不呼叫 shutdown 这个指令
- -i : 在关机之前先把所有网络相关的装置先停止
- -p : 当关机的时候，顺便做关闭电源（poweroff）的动作

实例

关闭系统

```
# halt
```

关闭系统并关闭电源

```
# halt -p
```

关闭系统，但不留下纪录

```
# halt -d
```

Linux nice命令

Linux nice命令以更改过的优先序来执行程序，如果未指定程序，则会印出目前的排程优先序，内定的 adjustment 为 10，范围为 -20（最高优先序）到 19（最低优先序）。

使用权限：所有使用者。

语法

```
nice [-n adjustment] [-adjustment] [--adjustment=adjustment] [--help] [--version] [comman
```

参数说明：

- -n adjustment, -adjustment, --adjustment=adjustment 皆为将该原有优先序的增加 adjustment
- --help 显示求助讯息
- --version 显示版本资讯

实例

设置程序运行时的优先级

```
# vi & //后台运行
[1] 15297
# nice vi & //设置默认优先级
[2] 15298

[1]+ Stopped         vi
# nice -n 19 vi & //设置优先级为19
[3] 15299

[2]+ Stopped         nice vi
# nice -n -20 vi & //设置优先级为 -20
[4] 15300

[3]+ Stopped         nice -n 19 vi
# ps -l //显示进程
F S  UID  PID PPID C PRI NI ADDR SZ WCHAN TTY     TIME CMD
4 S   0 15278 15212 0 80  0 - 1208 wait  pts/2  00:00:00 bash
0 T   0 15297 15278 0 80  0 - 2687 signal pts/2  00:00:00 vi
0 T   0 15298 15278 0 90 10 - 2687 signal pts/2  00:00:00 vi
0 T   0 15299 15278 1 99 19 - 2687 signal pts/2  00:00:00 vi
4 T   0 15300 15278 3 60 -20 - 2687 signal pts/2  00:00:00 vi
4 R   0 15301 15278 0 80  0 -  625 -   pts/2  00:00:00 ps

[4]+ Stopped         nice -n -20 vi
```

将 ls 的优先序加 1 并执行

```
nice -n 1 ls
```

将 ls 的优先序加 10 并执行

```
nice ls
```

注意：优先序 (priority) 为操作系统用来决定 CPU 分配的参数，Linux 使用『回合制(round-robin)』的演算法来做 CPU 排程，优先序越高，所可能获得的 CPU时间就越多。

Linux procinfo命令

Linux procinfo命令用于显示系统状态。

procinfo(process information)指令从/proc目录里读取相关数据，将数据妥善整理过后输出到标准输出设备。

语法

```
procinfo [-abdDfhimsSv][-F <输出文件>][-n <间隔秒数>]
```

参数说明：

- -a　显示所有信息。
- -b　显示磁盘设备的区块数目，而非存取数目。
- -d　显示系统信息每秒间的变化差额，而非总和的数值。本参数必须配合"-f"参数使用
- -D　此参数效果和指定"-d"参数类似，但内存和交换文件的信息为总和数值。
- -f　进入全画面的互动式操作界面。
- -F<输出文件>　把信息状态输出到文件保存起来，而非预设的标准输出设备。
- -h　在线帮助。
- -i　显示完整的IRP列表。
- -m　显示系统模块和外围设备等相关信息。
- -n<间隔秒数>　设置全画面互动模式的信息更新速度，单位以秒计算。
- -s　显示系统的内存，磁盘空间，IRP和DMA等信息，此为预设值。
- -S　搭配参数"-d"或"-D"使用时，每秒都会更新信息，不论是否有使用参数"-n"。
- -v　显示版本信息。

实例

显示系统状态

```
# procinfo
```

Linux top命令

Linux top命令用于实时显示 process 的动态。

使用权限：所有使用者。

语法

```
top [-] [d delay] [q] [c] [S] [s] [i] [n] [b]
```

参数说明：

- d : 改变显示的更新速度，或是在交谈式指令列(interactive command)按 s
- q : 没有任何延迟的显示速度，如果使用者是有 superuser 的权限，则 top 将会以最高的优先序执行
- c : 切换显示模式，共有两种模式，一是只显示执行档的名称，另一种是显示完整的路径与名称S : 累积模式，会将己完成或消失的子行程 (dead child process) 的 CPU time 累积起来
- s : 安全模式，将交谈式指令取消, 避免潜在的危机
- i : 不显示任何闲置 (idle) 或无用 (zombie) 的行程
- n : 更新的次数，完成后将会退出 top
- b : 批次档模式，搭配 "n" 参数一起使用，可以用来将 top 的结果输出到档案内

实例

显示进程信息

```
# top
```

显示完整命令

```
# top -c
```

以批处理模式显示程序信息

```
# top -b
```

以累积模式显示程序信息

```
# top -S
```

设置信息更新次数

```
top -n 2
//表示更新两次后终止更新显示
```

设置信息更新时间

```
# top -d 3
//表示更新周期为3秒
```

显示指定的进程信息

```
# top -p 139
//显示进程号为139的进程信息，CPU、内存占用率等
```

显示更新十次后退出

```
top -n 10
```

使用者将不能利用交谈式指令来对行程下命令

```
top -s
```

将更新显示二次的结果输入到名称为 top.log 的档案里

```
top -n 2 -b < top.log
```

Linux pstree命令

Linux pstree命令将所有行程以树状图显示，树状图将会以 pid (如果有指定) 或是以 init 这个基本行程为根 (root)，如果有指定使用者 id，则树状图会只显示该使用者所拥有的行程。

使用权限：所有使用者。

语法

```
pstree [-a] [-c] [-h|-Hpid] [-l] [-n] [-p] [-u] [-G|-U] [pid|user]
```

或

```
pstree -V
```

参数说明：

- -a 显示该行程的完整指令及参数, 如果是被记忆体置换出去的行程则会加上括号
- -c 如果有重覆的行程名, 则分开列出（预设值是会在前面加上 *）

实例

显示进程的关系

```
pstree
init-+-amd
|-apmd
|-atd
|-httpd---10*[httpd]
%pstree -p
init(1)-+-amd(447)
|-apmd(105)
|-atd(339)
%pstree -c
init-+-amd
|-apmd
|-atd
|-httpd-+-httpd
| |-httpd
| |-httpd
| |-httpd
....
```

特别表明在运行的进程

```
# pstree -apnh //显示进程间的关系
```

同时显示用户名称

```
# pstree -u //显示用户名称
```

Linux shutdown命令

Linux shutdown命令可以用来进行关机程序，并且在关机以前传送讯息给所有使用者正在执行的程序，shutdown 也可以用来重开机。

使用权限：系统管理者。

语法

```
shutdown [-t seconds] [-rkhncfF] time [message]
```

参数说明：

- -t seconds : 设定在几秒钟之后进行关机程序
- -k : 并不会真的关机，只是将警告讯息传送给所有只用者
- -r : 关机后重新开机
- -h : 关机后停机
- -n : 不采用正常程序来关机，用强迫的方式杀掉所有执行中的程序后自行关机
- -c : 取消目前已经进行中的关机动作
- -f : 关机时，不做 fcsk 动作(检查 Linux 档系统)
- -F : 关机时，强迫进行 fsck 动作
- time : 设定关机的时间
- message : 传送给所有使用者的警告讯息

实例

立即关机

```
# shutdown -h now
```

指定5分钟后关机

```
# shutdown +5 “System will shutdown after 5 minutes” //5分钟够关机并显示警告信息
```

Linux screen命令

Linux screen命令用于多重视窗管理程序。

screen为多重视窗管理程序。此处所谓的视窗，是指一个全屏幕的文字模式画面。通常只有在使用telnet登入主机或是使用老式的终端机时，才有可能用到screen程序。

语法

```
screen [-AmRvx -ls -wipe][-d <作业名称>][-h <行数>][-r <作业名称>][-s <shell>][-S <作业名称>]
```

参数说明：

- -A 将所有的视窗都调整为目前终端机的大小。
- -d<作业名称> 将指定的screen作业离线。
- -h<行数> 指定视窗的缓冲区行数。
- -m 即使目前已在作业中的screen作业，仍强制建立新的screen作业。
- -r<作业名称> 恢复离线的screen作业。
- -R 先试图恢复离线的作业。若找不到离线的作业，即建立新的screen作业。
- -s<shell> 指定建立新视窗时，所要执行的shell。
- -S<作业名称> 指定screen作业的名称。
- -v 显示版本信息。
- -x 恢复之前离线的screen作业。
- -ls或--list 显示目前所有的screen作业。
- -wipe 检查目前所有的screen作业，并删除已经无法使用的screen作业。

实例

创建 screen 终端

```
# screen //创建 screen 终端
```

创建 screen 终端 并执行任务

```
# screen vi ~/main.c //创建 screen 终端 ，并执行 vi命令
```

离开 screen 终端

```
# screen vi ~/main.c //创建 screen 终端 ，并执行 vi命令

#include

main ()
{

}

"~/mail.c"       0,0-1

在 screen 终端 下 按下 Ctrl+a d键
```

重新连接离开的 screen 终端

```
# screen -ls  //显示已创建的screen终端
There are screens on:
2433.pts-3.linux    (2013年10月20日 16时48分59秒)    (Detached)
2428.pts-3.linux    (2013年10月20日 16时48分05秒)    (Detached)
2284.pts-3.linux    (2013年10月20日 16时14分55秒)    (Detached)
2276.pts-3.linux    (2013年10月20日 16时13分18秒)    (Detached)
4 Sockets in /var/run/screen/S-root.

# screen -r 2276 //连接 screen_id 为 2276 的 screen终端
```

Linux sliplogin命令

Linux sliplogin命令用于将SLIP接口加入标准输入。

sliplogin可将SLIP接口加入标准输入，把一般终端机的连线变成SLIP连线。通常可用来建立SLIP服务器，让远端电脑以SLIP连线到服务器。sliplogin活去检查/etc/slip/slip.hosts文件中是否有相同的用户名称。通过检查后，sliplogin会调用执行shell script来设置IP地址，子网掩码等网络界面环境。此shell script通常是/etc/slip/slip.login。

语法

```
sliplogin [用户名称]
```

实例

改变用户的连接方式

```
# sliplogin kk // 改变用户的连接方式
```

Linux rsh命令

Linux rsh命令用于远端登入的Shell。

rsh(remote shell)提供用户环境，也就是Shell，以便指令能够在指定的远端主机上执行。

语法

```
rsh [-dn][-l <用户名称>][主机名称或IP地址][执行指令]
```

参数说明：

- -d 使用Socket层级的排错功能。
- -l<用户名称> 指定要登入远端主机的用户名称。
- -n 把输入的指令号向代号为/dev/null的特殊外围设备。

实例

开启rsh服务

```
# chkconfig --list //检测rlogin服务是否开启

# chkconfig rsh on //开启rsh服务

# chkconfig -list //检测开启的服务
```

远程命令执行

```
# rsh -l hnlinux 192.168.1.88 /bin/ls //远程执行ls命令
```

Linux rwho命令

Linux rwho命令用于查看系统用户。

rwho指令的效果类似who指令，但它会显示局域网里所有主机的用户。主机必须提供rwhod常驻服务的功能，方可使用rwho指令。

语法

```
rwho [-a]
```

参数说明：

- -a 列出所有的用户，包括闲置时间超过1个小时以上的用户。

实例

显示本地局域网内的所有用户

```
# rwho
root   snail-hnlinux:pts/2 May 14 17:42
```

Linux sudo命令

Linux sudo命令以系统管理者的身份执行指令，也就是说，经由 sudo 所执行的指令就好像是 root 亲自执行。

使用权限：在 /etc/sudoers 中有出现的使用者。

语法

```
sudo -V
```

```
sudo -h
```

```
sudo -l
```

```
sudo -v
```

```
sudo -k
```

```
sudo -s
```

```
sudo -H
```

```
sudo [ -b ] [ -p prompt ] [ -u username/#uid] -s
```

```
sudo command
```

参数说明：

- -V 显示版本编号
- -h 会显示版本编号及指令的使用方式说明
- -l 显示出自己（执行 sudo 的使用者）的权限
- -v 因为 sudo 在第一次执行时或是在 N 分钟内没有执行（N 预设为五）会问密码，这个参数是重新做一次确认，如果超过 N 分钟，也会问密码
- -k 将会强迫使用者在下一次执行 sudo 时问密码（不论有没有超过 N 分钟）
- -b 将要执行的指令放在背景执行

- -p prompt 可以更改问密码的提示语，其中 %u 会代换为使用者的帐号名称， %h 会显示主机名称
- -u username/#uid 不加此参数，代表要以 root 的身份执行指令，而加了此参数，可以以 username 的身份执行指令（#uid 为该 username 的使用者号码）
- -s 执行环境变数中的 SHELL 所指定的 shell ，或是 /etc/passwd 里所指定的 shell
- -H 将环境变数中的 HOME （家目录）指定为要变更身份的使用者家目录（如不加 -u 参数就是系统管理者 root ）
- command 要以系统管理者身份（或以 -u 更改为其他人）执行的指令

实例

sudo命令使用

```
$ sudo ls
[sudo] password for hnlinux:
hnlinux is not in the sudoers file. This incident will be reported.
```

指定用户执行命令

```
# sudo -u userb ls -l
```

显示sudo设置

```
$ sudo -L //显示sudo设置
Available options in a sudoers ``Defaults'' line:

syslog: Syslog facility if syslog is being used for logging
syslog_goodpri: Syslog priority to use when user authenticates successfully
syslog_badpri: Syslog priority to use when user authenticates unsuccessfully
long_otp_prompt: Put OTP prompt on its own line
ignore_dot: Ignore '.' in $PATH
mail_always: Always send mail when sudo is run
mail_badpass: Send mail if user authentication fails
mail_no_user: Send mail if the user is not in sudoers
mail_no_host: Send mail if the user is not in sudoers for this host
mail_no_perms: Send mail if the user is not allowed to run a command
tty_tickets: Use a separate timestamp for each user/tty combo
lecture: Lecture user the first time they run sudo
lecture_file: File containing the sudo lecture
authenticate: Require users to authenticate by default
root_sudo: Root may run sudo
log_host: Log the hostname in the (non-syslog) log file
log_year: Log the year in the (non-syslog) log file
shell_noargs: If sudo is invoked with no arguments, start a shell
set_home: Set $HOME to the target user when starting a shell with -s
always_set_home: Always set $HOME to the target user's home directory
path_info: Allow some information gathering to give useful error messages
fqdn: Require fully-qualified hostnames in the sudoers file
insults: Insult the user when they enter an incorrect password
requiretty: Only allow the user to run sudo if they have a tty
env_editor: Visudo will honor the EDITOR environment variable
rootpw: Prompt for root's password, not the users's
runaspw: Prompt for the runas_default user's password, not the users's
targetpw: Prompt for the target user's password, not the users's
use_loginclass: Apply defaults in the target user's login class if there is one
set_logname: Set the LOGNAME and USER environment variables
```

```
stay_setuid: Only set the effective uid to the target user, not the real uid
preserve_groups: Don't initialize the group vector to that of the target user
loglinelen: Length at which to wrap log file lines (0 for no wrap)
timestamp_timeout: Authentication timestamp timeout
passwd_timeout: Password prompt timeout
passwd_tries: Number of tries to enter a password
umask: Umask to use or 0777 to use user's
logfile: Path to log file
mailerpath: Path to mail program
mailerflags: Flags for mail program
mailto: Address to send mail to
mailfrom: Address to send mail from
mailsub: Subject line for mail messages
badpass_message: Incorrect password message
timestampdir: Path to authentication timestamp dir
timestampowner: Owner of the authentication timestamp dir
exempt_group: Users in this group are exempt from password and PATH requirements
passprompt: Default password prompt
passprompt_override: If set, passprompt will override system prompt in all cases.
runas_default: Default user to run commands as
secure_path: Value to override user's $PATH with
editor: Path to the editor for use by visudo
listpw: When to require a password for 'list' pseudocommand
verifypw: When to require a password for 'verify' pseudocommand
noexec: Preload the dummy exec functions contained in 'noexec_file'
noexec_file: File containing dummy exec functions
ignore_local_sudoers: If LDAP directory is up, do we ignore local sudoers file
closefrom: File descriptors >= %d will be closed before executing a command
closefrom_override: If set, users may override the value of `closefrom' with the -C optio
setenv: Allow users to set arbitrary environment variables
env_reset: Reset the environment to a default set of variables
env_check: Environment variables to check for sanity
env_delete: Environment variables to remove
env_keep: Environment variables to preserve
role: SELinux role to use in the new security context
type: SELinux type to use in the new security context
askpass: Path to the askpass helper program
env_file: Path to the sudo-specific environment file
sudoers_locale: Locale to use while parsing sudoers
visiblepw: Allow sudo to prompt for a password even if it would be visisble
pwfeedback: Provide visual feedback at the password prompt when there is user input
fast_glob: Use faster globbing that is less accurate but does not access the filesystem
umask_override: The umask specified in sudoers will override the user's, even if it is mo
```

以root权限执行上一条命令

```
$ sudo !!
```

以特定用户身份进行编辑文本

```
$ sudo -u uggc vi ~www/index.html
//以 uggc 用户身份编辑  home 目录下www目录中的 index.html 文件
```

列出目前的权限

```
sudo -l
```

列出 sudo 的版本资讯

```
sudo -V
```

Linux gitps命令

Linux gitps命令用于报告程序状况。

gitps(gnu interactive tools process status)是用来报告并管理程序执行的指令，基本上它就是通过ps指令来报告，管理程序，也能通过gitps指令随时中断，删除不必要的程序。因为gitps指令会去执行ps指令，所以其参数和ps指令相当类似。

语法

```
gitps [acefgjlnrsSTuvwxX][p <程序识别码>][t <终端机编号>][U <帐号名称>]
```

参数说明：

- a 显示 现行终端机下的所有程序，包括其他用户的程序。
- c 列出程序时，显示每个程序真正的指令名称，而不包含路径，参数或是常驻服务的标示.
- e 列出程序时，显示每个程序所使用的环境变量。
- f 用ASCII字符显示树状结构，表达程序间的相互关系。
- g 显示现行终端机下的所有程序，包括群组领导者的程序。
- j 采用工作控制的格式来显示程序状况。
- l 采用纤细的格式来显示程序状况。
- n 以数字来表示USER和WCHAN栏位。
- p<程序识别码> 指定程序识别码，并列出该程序的状况。
- r 只列出现行终端机正在执行中的程序。
- s 采用程序信号的格式显示程序状况。
- S 列出程序时，包括已中断的子程序信息。
- t<终端机机标号> 指定终端机编号，并列出属于该终端机的程序的状况。
- T 显示现行终端机下的所有程序。
- u 以用户为主的格式来显示程序状况。
- U<帐号名称> 列出属于该用户的程序的状况。
- v 采用虚拟内存的格式显示程序状况。
- w 采用宽阔的格式来显示程序状况。
- x 显示所有程序，不以终端机来区分。
- X 采用旧试的Linux i386登陆格式显示程序状况。

实例

显示指定用户信息

```
# gitps hnlinux
```

Linux uname命令

Linux uname命令用于显示系统信息。

uname可显示电脑以及操作系统的相关信息。

语法

```
uname [-amnrsv][--help][--version]
```

参数说明：

- -a或--all 显示全部的信息。
- -m或--machine 显示电脑类型。
- -n或-nodename 显示在网络上的主机名称。
- -r或--release 显示操作系统的发行编号。
- -s或--sysname 显示操作系统名称。
- -v 显示操作系统的版本。
- --help 显示帮助。
- --version 显示版本信息。

实例

显示系统信息

```
# uname -a
Linux snail-hnlinux 2.6.32-21-generic #32-Ubuntu SMP Fri Apr 16 08:10:02 UTC 2010 i686 GN
```

显示计算机类型

```
# uname -m
i686
```

显示计算机名

```
# uname -n
snail-hnlinux
```

显示操作系统发行编号

```
# uname -r
2.6.32-21-generic
```

显示操作系统名称

```
# uname -s
Linux
```

显示系统时间

```
# uname -v
#32-Ubuntu SMP Fri Apr 16 08:10:02 UTC 2014
```

Linux logrotate命令

Linux logrotate命令用于管理记录文件。

使用logrotate指令，可让你轻松管理系统所产生的记录文件。它提供自动替换，压缩，删除和邮寄记录文件，每个记录文件都可被设置成每日，每周或每月处理，也能在文件太大时立即处理。您必须自行编辑，指定配置文件，预设的配置文件存放在/etc目录下，文件名称为logrotate.conf。

语法

```
logrotate [-?dfv][-s <状态文件>][--usage][配置文件]
```

参数说明：

- -?或--help　在线帮助。
- -d或--debug　详细显示指令执行过程，便于排错或了解程序执行的情况。
- -f或--force　强行启动记录文件维护操作，纵使logrotate指令认为没有需要亦然。
- -s<状态文件>或--state=<状态文件>　使用指定的状态文件。
- -v或--version　显示指令执行过程。
- -usage　显示指令基本用法。

实例

指定记录文件

```
# logrotate /root/log.config
```

Linux tload命令

Linux tload命令用于显示系统负载状况。

tload指令使用ASCII字符简单地以文字模式显示系统负载状态。假设不给予终端机编号，则会在执行tload指令的终端机显示负载情形。

语法

```
tload [-V][-d <间隔秒数>][-s <刻度大小>][终端机编号]
```

参数说明：

- -d<间隔秒数>　设置tload检测系统负载的间隔时间，单位以秒计算。
- -s<刻度大小>　设置图表的垂直刻度大小，单位以列计算。
- -V　显示版本信息。

实例

显示系统负载

```
# tload
```

Linux swatch命令

Linux swatch命令用于系统监控程序。

swatch可用来监控系统记录文件，并在发现特定的事件时，执行指定的动作。swatch所监控的事件以及对应事件的动作都存放在swatch的配置文件中。预设的配置文件为拥护根目录下的.swatchrc。然而在Red Hat Linux的预设用户根目录下并没有.swatchrc配置文件，您可将/usr/doc/swatch-2.2/config_files/swatchrc.personal文件复制到用户根目录下的.swatchrc，然后修改.swatchrc所要监控的事件及执行的动作。

语法

```
swatch [-A <分隔字符>][-c <设置文件>][-f <记录文件>][-I <分隔字符>][-P <分隔字符>][-r <时间>][-t
```

参数说明：

- -A<分隔字符> 预设配置文件中，动作的分隔字符，预设为逗号。
- -c<设置文件> 指定配置文件，而不使用预设的配置文件。
- -f<记录文件> 检查指定的记录文件，检查完毕后不会继续监控该记录文件。
- -I<分隔字符> 指定输入记录的分隔字符，预设为换行字符。
- -P<分隔字符> 指定配置文件中，事件的分隔字符，预设为逗号。
- -r<时间> 在指定的时间重新启动。
- -t<记录文件> 检查指定的记录文件，并且会监控加入记录文件中的后继记录。

实例

开启系统监视

```
# swatch
```

Linux chsh命令

Linux chsh命令用于更改使用者 shell 设定。

使用权限：所有使用者。

语法

```
shell>> chsh
```

实例

```
shell>> chsh
Changing fihanging shell for user1
Password: [del]
New shell [/bin/tcsh]: ### [是目前使用的 shell]
[del]
shell>> chsh -l ### 展示 /etc/shells 档案内容
/bin/bash
/bin/sh
/bin/ash
/bin/bsh
/bin/tcsh
/bin/csh
```

改变当前的shell。当前的shell 设置为//bin/bash，通过chsh命令，改变shell的设置/bin/csh。

```
# chsh
Changing shell for root.
New shell [/bin/bash]: /bin/csh //输入新的shell地址
Shell changed.
```

通过 -s 参数改变当前的shell设置

```
# chsh -s /bin/csh //改变当前设置为 /bin/csh
Changing shell for root.
Shell not changed.
```

Linux whoami命令

Linux whoami命令用于显示自身用户名称。

显示自身的用户名称，本指令相当于执行"id -un"指令。

语法

```
whoami [--help][--version]
```

参数说明：

- --help 　在线帮助。
- --version 　显示版本信息。

实例

显示用户名

```
# whoami
root
```

Linux who命令

Linux who命令用于显示系统中有哪些使用者正在上面，显示的资料包含了使用者 ID、使用的终端机、从哪边连上来的、上线时间、呆滞时间、CPU 使用量、动作等等。

使用权限：所有使用者都可使用。

语法

```
who - [husfV] [user]
```

参数说明：

- -h : 不要显示标题列
- -u : 不要显示使用者的动作/工作
- -s : 使用简短的格式来显示
- -f : 不要显示使用者的上线位置
- -V : 显示程序版本

实例

显示当前登录系统的用户

```
# who  //显示当前登录系统的用户
root   tty7     2014-05-13 12:12 (:0)
root   pts/0    2014-05-14 17:09 (:0.0)
root   pts/1    2014-05-14 18:51 (192.168.1.17)
root   pts/2    2014-05-14 19:48 (192.168.1.17)
```

显示标题栏

```
# who -H
NAME   LINE     TIME        COMMENT
root   tty7     2014-05-13 12:12 (:0)
root   pts/0    2014-05-14 17:09 (:0.0)
root   pts/1    2014-05-14 18:51 (192.168.1.17)
root   pts/2    2014-05-14 19:48 (192.168.1.17)
```

显示用户登录来源

```
# who -l -H
NAME   LINE     TIME        IDLE     PID COMMENT
LOGIN  tty4     2014-05-13 12:11         852 id=4
LOGIN  tty5     2014-05-13 12:11         855 id=5
LOGIN  tty2     2014-05-13 12:11         862 id=2
LOGIN  tty3     2014-05-13 12:11         864 id=3
LOGIN  tty6     2014-05-13 12:11         867 id=6
LOGIN  tty1     2014-05-13 12:11        1021 id=1
```

显示终端属性

```
# who -T -H
NAME    LINE     TIME        COMMENT
root   + tty7     2014-05-13 12:12 (:0)
root   + pts/0    2014-05-14 17:09 (:0.0)
root   - pts/1    2014-05-14 18:51 (192.168.1.17)
root   - pts/2    2014-05-14 19:48 (192.168.1.17)
```

只显示当前用户

```
# who -m -H
NAME   LINE     TIME        COMMENT
root   pts/1    2014-05-14 18:51 (192.168.1.17)
```

精简模式显示

```
# who -q
root root root root
# users=4
```

Linux vlock命令

Linux vlock命令用于锁住虚拟终端。

执行vlock(virtual console lock)指令可锁住虚拟终端，避免他人使用。

语法

```
vlock [-achv]
```

参数说明：

- -a或--all 　锁住所有的终端阶段作业，如果您在全屏幕的终端中使用本参数，则会将用键盘
- 切换终端机的功能一并关闭。
- -c或--current 　锁住目前的终端阶段作业，此为预设值。
- -h或--help 　在线帮助。
- -v或--version 　显示版本信息。

实例

锁定虚拟终端

```
# vlock
```

Linux usermod命令

Linux usermod命令用于修改用户帐号。

usermod可用来修改用户帐号的各项设定。

语法

```
usermod [-LU][-c <备注>][-d <登入目录>][-e <有效期限>][-f <缓冲天数>][-g <群组>][-G <群组>][-l
```

参数说明：

- -c<备注>　修改用户帐号的备注文字。
- -d登入目录>　修改用户登入时的目录。
- -e<有效期限>　修改帐号的有效期限。
- -f<缓冲天数>　修改在密码过期后多少天即关闭该帐号。
- -g<群组>　修改用户所属的群组。
- -G<群组>　修改用户所属的附加群组。
- -l<帐号名称>　修改用户帐号名称。
- -L　锁定用户密码，使密码无效。
- -s<shell>　修改用户登入后所使用的shell。
- -u<uid>　修改用户ID。
- -U　解除密码锁定。

实例

更改登录目录

```
# usermod -d /home/hnlinux root
```

改变用户的uid

```
# usermod -u 777 root
```

Linux userdel命令

Linux userdel命令用于删除用户帐号。

userdel可删除用户帐号与相关的文件。若不加参数，则仅删除用户帐号，而不删除相关文件。

语法

```
userdel [-r][用户帐号]
```

参数说明：

- -r 　删除用户登入目录以及目录中所有文件。

实例

删除用户账号

```
# userdel hnlinux
```

Linux userconf命令

Linux userconf命令用于用户帐号设置程序。

userconf实际上为linuxconf的符号连接，提供图形界面的操作方式，供管理员建立与管理各类帐号。若不加任何参数，即进入图形界面。

语法

```
userconf [--addgroup <群组>][--adduser <用户ID><群组><用户名称><shell>][--delgroup <群组>][--
```

参数说明：

- --addgroup<群组>　新增群组。
- --adduser<用户ID><群组><用户名称><shell>　新增用户帐号。
- --delgroup<群组>　删除群组。
- --deluser<用户ID>　删除用户帐号。
- --help　显示帮助。

实例

新增用户

```
# userconf --adduser 666 tt lord /bin/bash //新增用户账号
```

Linux id命令

Linux id命令用于显示用户的ID，以及所属群组的ID。

id会显示用户以及所属群组的实际与有效ID。若两个ID相同，则仅显示实际ID。若仅指定用户名称，则显示目前用户的ID。

语法

```
id [-gGnru][--help][--version][用户名称]
```

参数说明：

- -g或--group 显示用户所属群组的ID。
- -G或--groups 显示用户所属附加群组的ID。
- -n或--name 显示用户，所属群组或附加群组的名称。
- -r或--real 显示实际ID。
- -u或--user 显示用户ID。
- -help 显示帮助。
- -version 显示版本信息。

实例

显示当前用户信息

```
# id //显示当前用户ID
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm),6(disk),10(wheel) c
```

显示用户群组的ID

```
# id -g
0
```

显示所有群组的ID

```
# id -g
0 1 2 3 4 5 6 10
```

显示指定用户信息

```
# id hnlinux
```

Linux w命令

Linux w命令用于显示目前登入系统的用户信息。

执行这项指令可得知目前登入系统的用户有哪些人，以及他们正在执行的程序。

单独执行 w 指令会显示所有的用户，您也可指定用户名称，仅显示某位用户的相关信息。

语法

```
w [-fhlsuV][用户名称]
```

参数说明：

- -f 开启或关闭显示用户从何处登入系统。
- -h 不显示各栏位的标题信息列。
- -l 使用详细格式列表，此为预设值。
- -s 使用简洁格式列表，不显示用户登入时间，终端机阶段作业和程序所耗费的CPU时间。
- -u 忽略执行程序的名称，以及该程序耗费CPU时间的信息。
- -V 显示版本信息。

实例

显示当前用户

```
w  //显示当前用户，不显示登录位置
19:50:14 up 9:27, 4 users, load average: 0.31, 0.26, 0.18
USER   TTY   FROM       LOGIN@  IDLE  JCPU  PCPU WHAT
root   tty7   :0        Thu12  31:39m 10:10  0.60s gnome-session
root   pts/0  :0.0       17:09  2:18m 15.26s 0.15s bash
root   pts/1  192.168.1.17   18:51  1.00s 1.24s 0.14s -bash
root   pts/2  192.168.1.17   19:48  60.00s 0.05s 0.05s -bash
```

不显示登录位置

```
w -f
19:53:59 up 9:31, 4 users, load average: 0.05, 0.16, 0.15
USER   TTY    LOGIN@  IDLE  JCPU  PCPU WHAT
root   tty7   Thu12  31:43m 10:10  0.60s gnome-session
root   pts/0   17:09  2:21m 15.26s 0.15s bash
root   pts/1   18:51  0.00s 1.04s 0.14s -bash
root   pts/2   19:48  4:45  0.05s 0.05s -bash
```

以精简模式显示

```
w -s
19:54:37 up 9:31, 4 users, load average: 0.24, 0.19, 0.16
USER   TTY   FROM        IDLE WHAT
root   tty7   :0        31:43m gnome-session
root   pts/0  :0.0       2:22m bash
root   pts/1  192.168.1.17   0.00s -bash
root   pts/2  192.168.1.17   5:23 -bash
```

不显示标题

```
w -h
root   tty7   :0        Thu12  31:44m 10:10  0.60s gnome-session
root   pts/0  :0.0       17:09  2:23m 15.26s 0.15s bash
root   pts/1  192.168.1.17   18:51  0.00s 1.05s 0.14s -bash
root   pts/2  192.168.1.17   19:48  5:54  0.05s 0.05s -bash
```

Linux skill命令

Linux skill命令送个讯号给正在执行的程序，预设的讯息为 TERM (中断)，较常使用的讯息为 HUP、INT、KILL、STOP、CONT 和 0。

讯息有三种写法：分别为 -9、-SIGKILL、-KILL，可以使用 -l 或 -L 已列出可使用的讯息。

使用权限：所有使用者。

其他相关的命令：kill

语法

```
skill [signal to send] [options] 选择程序的规则
```

一般参数：

- -f 快速模式/尚未完成
- -i 互动模式/ 每个动作将要被确认
- -v 详细输出/ 列出所选择程序的资讯
- -w 智能警告讯息/ 尚未完成
- -n 没有动作/ 显示程序代号

参数：选择程序的规则可以是：终端机代号、使用者名称、程序代号、命令名称。

- -t 终端机代号 (tty 或 pty)
- -u 使用者名称
- -p 程序代号 (pid)
- -c 命令名称可使用的讯号

以下列出已知的讯号名称、讯号代号、功能。

名称（代号）	功能/描述
ALRM 14	离开
HUP 1	离开
INT 2	离开
KILL 9	离开/强迫关闭
PIPE 13	离开
POLL	离开
PROF	离开

TERM 15	离开
USR1	离开
USR2	离开
VTALRM	离开
STKFLT	离开/只适用于i386、m68k、arm 和 ppc 硬件
UNUSED	离开/只适用于i386、m68k、arm 和 ppc 硬件
TSTP	停止/产生与内容相关的行为
TTIN	停止/产生与内容相关的行为
TTOU	停止/产生与内容相关的行为
STOP	停止/强迫关闭
CONT	重新启动/如果在停止状态则重新启动，否则忽略
PWR	忽略/在某些系统中会离开
WINCH	忽略
CHLD	忽略
ABRT 6	核心
FPE 8	核心
ILL 4	核心
QUIT 3	核心
SEGV 11	核心
TRAP 5	核心
SYS	核心/或许尚未实作
EMT	核心/或许尚未实作
BUS	核心/核心失败
XCPU	核心/核心失败
XFSZ	核心/核心失败

实例

停止所有在 PTY 装置上的程序

```
skill -KILL -v pts/*
```

停止三个使用者 user1、user2、user3

```
skill -STOP user1 user2 user3
```

Linux su命令

Linux su命令用于变更为其他使用者的身份，除 root 外，需要键入该使用者的密码。

使用权限：所有使用者。

语法

```
su [-fmp] [-c command] [-s shell] [--help] [--version] [-] [USER [ARG]]
```

参数说明：

- -f 或 --fast 不必读启动档（如 csh.cshrc 等），仅用于 csh 或 tcsh
- -m -p 或 --preserve-environment 执行 su 时不改变环境变数
- -c command 或 --command=command 变更为帐号为 USER 的使用者并执行指令（command）后再变回原来使用者
- -s shell 或 --shell=shell 指定要执行的 shell （bash csh tcsh 等），预设值为 /etc/passwd 内的该使用者（USER） shell
- --help 显示说明文件
- --version 显示版本资讯
-
    - -l 或 --login 这个参数加了之后，就好像是重新 login 为该使用者一样，大部份环境变数（HOME SHELL USER等等）都是以该使用者（USER）为主，并且工作目录也会改变，如果没有指定 USER ，内定是 root
- USER 欲变更的使用者帐号
- ARG 传入新的 shell 参数

实例

变更帐号为 root 并在执行 ls 指令后退出变回原使用者

```
su -c ls root
```

变更帐号为 root 并传入 -f 参数给新执行的 shell

```
su root -f
```

变更帐号为 clsung 并改变工作目录至 clsung 的家目录（home dir）

```
su - clsung
```

切换用户

```
hnlinux@w3cschool.cc:~$ whoami //显示当前用户
hnlinux
hnlinux@w3cschool.cc:~$ pwd //显示当前目录
/home/hnlinux
hnlinux@w3cschool.cc:~$ su root //切换到root用户
密码：
root@w3cschool.cc:/home/hnlinux# whoami
root
root@w3cschool.cc:/home/hnlinux# pwd
/home/hnlinux
```

切换用户，改变环境变量

```
hnlinux@w3cschool.cc:~$ whoami //显示当前用户
hnlinux
hnlinux@w3cschool.cc:~$ pwd //显示当前目录
/home/hnlinux
hnlinux@w3cschool.cc:~$ su - root //切换到root用户
密码：
root@w3cschool.cc:/home/hnlinux# whoami
root
root@w3cschool.cc:/home/hnlinux# pwd //显示当前目录
/root
```

Linux renice命令

Linux renice命令用于重新指定一个或多个行程（Process）的优先序（一个或多个将根据参数而定）。

注意：每一个行程（Process）都有一个唯一的（unique）id。

使用权限：所有使用者。

语法

```
renice priority [[-p] pid ...] [[-g] pgrp ...] [[-u] user ...]
```

参数说明：

- -p pid 重新指定行程的 id 为 pid 的行程的优先序
- -g pgrp 重新指定行程群组(process group)的 id 为 pgrp 的行程 (一个或多个) 的优先序
- -u user 重新指定行程拥有者为 user 的行程的优先序

实例

将行程 id 为 987 及 32 的行程与行程拥有者为 daemon 及 root 的优先序号码加 1

```
renice +1 987 -u daemon root -p 32
```

Linux newgrp命令

Linux newgrp命令用于登入另一个群组。

newgrp指令类似login指令，当它是以相同的帐号，另一个群组名称，再次登入系统。欲使用newgrp指令切换群组，您必须是该群组的用户，否则将无法登入指定的群组。单一用户要同时隶属多个群组，需利用交替用户的设置。若不指定群组名称，则newgrp指令会登入该用户名称的预设群组。

语法

```
newgrp [群组名称]
```

实例

改变群组

```
# newgrp root
```

Linux whois命令

Linux whois命令用于查找并显示用户信息。

whois指令会去查找并显示指定帐号的用户相关信息，因为它是到Network Solutions的WHOIS数据库去查找，所以该帐号名称必须在上面注册方能寻获，且名称没有大小写的差别。

语法

```
whois [帐号名称]
```

实例

显示指定用户信息

```
# whois root

//查找root用户信息
```

查询域名描述信息

```
# whois .Lx138.COm

Whois Server Version 2.0

Domain names in the .com and .net domains can now be registered
with many different competing registrars. Go to http://www.internic.net
for detailed information.

...省略部分内容
```

查询域名信息

```
# whois Lx138.COm

The Registry database contains ONLY .COM, .NET, .EDU domains and
Registrars.
Domain Name ..................... Lx138.COm
Name Server ..................... dns15.hichina.com
                 dns16.hichina.com
Registrant ID ................... hc937242545-cn

...省略部分内容
```

查询域名信息省略法律声明

```
# whois -H Lx138.COm

...省略内容
```

指定端口查询

```
# whois -p 80 Lx138.COm

...省略内容
```

Linux free命令

Linux free命令用于显示内存状态。

free指令会显示内存的使用情况，包括实体内存，虚拟的交换文件内存，共享内存区段，以及系统核心使用的缓冲区等。

语法

```
free [-bkmotV][-s <间隔秒数>]
```

参数说明：

- -b 　以Byte为单位显示内存使用情况。
- -k 　以KB为单位显示内存使用情况。
- -m 　以MB为单位显示内存使用情况。
- -o 　不显示缓冲区调节列。
- -s<间隔秒数> 　持续观察内存使用状况。
- -t 　显示内存总和列。
- -V 　显示版本信息。

实例

显示内存使用情况

```
# free //显示内存使用信息
total used free shared buffers cached
Mem: 254772 184568 70204 0 5692 89892
-/+ buffers/cache: 88984 165788
Swap: 524280 65116 459164
```

以总和的形式显示内存的使用信息

```
# free -t //以总和的形式查询内存的使用信息
total used free shared buffers cached
Mem: 254772 184868 69904 0 5936 89908
-/+ buffers/cache: 89024 165748
Swap: 524280 65116 459164
Total: 779052 249984 529068
```

周期性的查询内存使用信息

```
# free -s 10 //每10s 执行一次命令
total used free shared buffers cached
Mem: 254772 187628 67144 0 6140 89964
-/+ buffers/cache: 91524 163248
Swap: 524280 65116 459164

total used free shared buffers cached
Mem: 254772 187748 67024 0 6164 89940
-/+ buffers/cache: 91644 163128
Swap: 524280 65116 459164
```

Linux命令大全 - 系统设定

reset	clear	alias	dircolors
aumix	bind	chroot	clock
crontab	declare	depmod	dmesg
enable	eval	export	pwunconv
grpconv	rpm	insmod	kbdconfig
lilo	liloconfig	lsmod	minfo
set	modprobe	ntsysv	mouseconfig
passwd	pwconv	rdate	resize
rmmod	grpunconv	modinfo	time
setup	sndconfig	setenv	setconsole
timeconfig	ulimit	unset	chkconfig
apmd	hwclock	mkkickstart	fbset
unalias	SVGATextMode		

Linux bind命令

Linux bind命令用于显示或设置键盘按键与其相关的功能。

您可以利用bind命令了解有哪些按键组合与其功能，也可以自行指定要用哪些按键组合。

语法

```
bind [-dlv][-f <按键配置文件>][-m <按键配置>][-q <功能>]
```

参数说明：

- -d 显示按键配置的内容。
- -f<按键配置文件> 载入指定的按键配置文件。
- -l 列出所有的功能。
- -m<按键配置> 指定按键配置。
- -q<功能> 显示指定功能的按键。
- -v 列出目前的按键配置与其功能。

实例

显示按键组合的所有功能

```
# bind -l //显示按键组合的内容
abort
accept-line
alias-expand-line
arrow-key-prefix
backward-byte
backward-char
backward-delete-char
backward-kill-line
backward-kill-word
backward-word
beginning-of-history
beginning-of-line
……省略部分内容
vi-goto-mark
vi-insert-beg
vi-insertion-mode
vi-match
vi-movement-mode
vi-next-word
vi-overstrike
vi-overstrike-delete
vi-prev-word
vi-put
vi-redo
vi-replace
vi-rubout
vi-search
vi-search-again
vi-set-mark
vi-subst
vi-tilde-expand
vi-yank-arg
vi-yank-to
yank
yank-last-arg
yank-nth-arg
yank-pop
```

显示当前按键组合的设置

```
# bind -l
abort
accept-line
alias-expand-line
arrow-key-prefix
backward-byte
backward-char
backward-delete-char
backward-kill-line
backward-kill-word
backward-word
beginning-of-history
beginning-of-line
call-last-kbd-macro
capitalize-word
character-search
character-search-backward
clear-screen
complete
complete-command
complete-filename
complete-hostname
complete-into-braces
complete-username
complete-variable
copy-backward-word
copy-forward-word
```

```
copy-region-as-kill
dabbrev-expand
delete-char
delete-char-or-list
delete-horizontal-space
digit-argument
display-shell-version
do-lowercase-version
downcase-word
dump-functions
dump-macros
dump-variables
dynamic-complete-history
edit-and-execute-command
emacs-editing-mode
end-kbd-macro
end-of-history
end-of-line
exchange-point-and-mark
forward-backward-delete-char
forward-byte
forward-char
forward-search-history
forward-word
glob-complete-word
glob-expand-word
glob-list-expansions
history-and-alias-expand-line
history-expand-line
history-search-backward
history-search-forward
insert-comment
insert-completions
insert-last-argument
kill-line
kill-region
kill-whole-line
kill-word
magic-space
menu-complete
menu-complete-backward
next-history
non-incremental-forward-search-history
non-incremental-forward-search-history-again
non-incremental-reverse-search-history
non-incremental-reverse-search-history-again
old-menu-complete
operate-and-get-next
overwrite-mode
possible-command-completions
possible-completions
possible-filename-completions
possible-hostname-completions
possible-username-completions
possible-variable-completions
previous-history
quoted-insert
redraw-current-line
re-read-init-file
reverse-search-history
revert-line
self-insert
set-mark
shell-backward-kill-word
shell-backward-word
shell-expand-line
shell-forward-word
shell-kill-word
skip-csi-sequence
start-kbd-macro
tab-insert
tilde-expand
```

```
transpose-chars
transpose-words
tty-status
undo
universal-argument
unix-filename-rubout
unix-line-discard
unix-word-rubout
upcase-word
vi-append-eol
vi-append-mode
vi-arg-digit
vi-back-to-indent
vi-bword
vi-bWord
vi-change-case
vi-change-char
vi-change-to
vi-char-search
vi-column
vi-complete
vi-delete
vi-delete-to
vi-editing-mode
vi-end-word
vi-eof-maybe
vi-eword
vi-eWord
vi-fetch-history
vi-first-print
vi-fword
vi-fWord
vi-goto-mark
vi-insert-beg
vi-insertion-mode
vi-match
vi-movement-mode
vi-next-word
vi-overstrike
vi-overstrike-delete
vi-prev-word
vi-put
vi-redo
vi-replace
vi-rubout
vi-search
vi-search-again
vi-set-mark
vi-subst
vi-tilde-expand
vi-yank-arg
vi-yank-to
yank
yank-last-arg
yank-nth-arg
yank-pop
root@snail-hnlinux:~#
root@snail-hnlinux:~#
root@snail-hnlinux:~#
root@snail-hnlinux:~#
root@snail-hnlinux:~# bind -v
set bind-tty-special-chars on
set blink-matching-paren on
set byte-oriented off
set completion-ignore-case off
set convert-meta off
set disable-completion off
set echo-control-characters on
set enable-keypad off
set enable-meta-key on
set expand-tilde off
set history-preserve-point off
```

```
set horizontal-scroll-mode off
set input-meta on
set mark-directories on
set mark-modified-lines off
set mark-symlinked-directories off
set match-hidden-files on
set meta-flag on
set output-meta on
set page-completions on
set prefer-visible-bell on
set print-completions-horizontally off
set revert-all-at-newline off
set show-all-if-ambiguous off
set show-all-if-unmodified off
set skip-completed-text off
set visible-stats off
set bell-style audible
set comment-begin #
set completion-prefix-display-length 0
set completion-query-items 100
set editing-mode emacs
set history-size 1000
set keymap emacs
```

列出指定功能的按键和按键组合

```
# bind -q abort
//请用 调用abort “C-g”, “C-xC-g”, “eC-g”.

# bind -q accept-line //列出功能“accept-line”按键以及组合按键
//请用 调用accept-line “C-j”, “C-m”.
```

Linux aumix命令

Linux aumix命令用于设置音效装置。

aumix(audio mixer)命令设置各项音效装置的信号强度以及指定播放与录音的装置。

语法

```
aumix [-123bcilmoprstvwWx][(+/-)强度][PqR][-dfhILqS]
```

参数说明：[-123bcilmoprstvwWx]为频道参数，用来指定装置的频道；[PqR]可用来指定播放或录音装置；[-dfhILqS] 则为指令参数。若不加任何参数，aumix会显示简单的图形界面供调整设置频道参数。

- -1　输入信号线 1 。
- -2　输入信号线 2 。
- -3　输入信号线 3 。
- -b　低音。
- -c　CD。
- -i　输入信号强度。
- -m　麦克风。
- -o　输出信号强度。
- -p　PC喇叭。
- -r　录音。
- -s　合成器。
- -t　高音。
- -v　主音量。
- -w　PCM。
- -W　PCM2。
- -x　混音器。
- (+/-)强度　出现(+/-)时，代表在原有的强度上加减指定值。若未使用(+/-)，则直接将强度设为指定值。　指定音效装置
- P　指定播放装置。
- q　显示频道设置。
- R　指定录音装置。

指令参数：

- -d　指定音效装置的名称。
- -f　指定存储或载入设置的文件。

- -h　在使用时显示信息。
- -I　以图形界面方式来执行aumix。
- -L　从$HOME/.aumixrc或/etc/aumixrc载入设置。
- -q　显示所有频道的设置值。
- -S　将设置值保存至/HOME/.aumixrc。

实例

设置音效设备

```
# aumix
```

Linux dircolors命令

Linux dircolors命令用于设置 ls 指令在显示目录或文件时所用的色彩。

dircolors可根据[色彩配置文件]来设置LS_COLORS环境变量或是显示设置LS_COLORS环境变量的shell指令。

语法

```
dircolors [色彩配置文件]
```

或

```
dircolors [-bcp][--help][--version]
```

参数说明：

- -b或--sh或--bourne-shell 　显示在Boume shell中，将LS_COLORS设为目前预设置的shell指令。
- -c或--csh或--c-shell 　显示在C shell中，将LS_COLORS设为目前预设置的shell指令。
- -p或--print-database 　显示预设置
- -help 　显示帮助。
- -version 　显示版本信息。

实例

显示默认值

```
# dircolors -p //显示默认值
# Configuration file for dircolors, a utility to help you set the
# LS_COLORS environment variable used by GNU ls with the --color option.
# Copyright (C) 1996, 1999-2008
# Free Software Foundation, Inc.
# Copying and distribution of this file, with or without modification,
# are permitted provided the copyright notice and this notice are preserved.
# The keywords COLOR, OPTIONS, and EIGHTBIT (honored by the
# slackware version of dircolors) are recognized but ignored.
# Below, there should be one TERM entry for each termtype that is colorizable
TERM Eterm
TERM ansi
TERM color-xterm
TERM con132x25
TERM con132x30
TERM con132x43
TERM con132x60
TERM con80x25
TERM con80x28
TERM xterm-debian
```

```
# Below are the color init strings for the basic file types. A color init
# string consists of one or more of the following numeric codes:
# Attribute codes:
# 00=none 01=bold 04=underscore 05=blink 07=reverse 08=concealed
# Text color codes:
# 30=black 31=red 32=green 33=yellow 34=blue 35=magenta 36=cyan 37=white
# Background color codes:
# 40=black 41=red 42=green 43=yellow 44=blue 45=magenta 46=cyan 47=white
#NORMAL 00 # no color code at all
#FILE 00 # regular file: use no color at all
RESET 0 # reset to “normal“ color
DIR 01;34 # directory
LINK 01;36 # symbolic link. (If you set this to 'target' instead of a
# numerical value, the color is as for the file pointed to.)
HARDLINK 44;37 # regular file with more than one link
FIFO 40;33 # pipe
SOCK 01;35 # socket
DOOR 01;35 # door
BLK 40;33;01 # block device driver
CHR 40;33;01 # character device driver
ORPHAN 40;31;01 # symlink to nonexistent file, or non-stat'able file
SETUID 37;41 # file that is setuid (u+s)
SETGID 30;43 # file that is setgid (g+s)
CAPABILITY 30;41 # file with capability
STICKY_OTHER_WRITABLE 30;42 # dir that is sticky and other-writable (+t,o+w)
OTHER_WRITABLE 34;42 # dir that is other-writable (o+w) and not sticky
STICKY 37;44 # dir with the sticky bit set (+t) and not other-writable
# This is for files with execute permission:
EXEC 01;32
# List any file extensions like '.gz' or '.tar' that you would like ls
# to colorize below. Put the extension, a space, and the color init string.
# (and any comments you want to add after a '#')
# If you use DOS-style suffixes, you may want to uncomment the following:
#.cmd 01;32 # executables (bright green)
#.exe 01;32
#.com 01;32
#.btm 01;32
#.bat 01;32
# Or if you want to colorize scripts even if they do not have the
# executable bit actually set.
#.sh 01;32
#.csh 01;32
# archives or compressed (bright red)
.tar 01;31

.pcx 01;35
.mov 01;35
.mpg 01;35
.mpeg 01;35
.m2v 01;35
.mkv 01;35
.ogm 01;35
.mp4 01;35
.m4v 01;35
.mp4v 01;35
.vob 01;35
.qt 01;35
.nuv 01;35
.wmv 01;35
.asf 01;35
.rm 01;35
.rmvb 01;35
.flc 01;35
.avi 01;35
.fli 01;35
.flv 01;35
.gl 01;35
.dl 01;35
.xcf 01;35
.xwd 01;35
.yuv 01;35
# http://wiki.xiph.org/index.php/MIME_Types_and_File_Extensions
```

```
.axv 01;35
.anx 01;35
.ogv 01;35
.ogx 01;35
# audio formats
.aac 00;36
.au 00;36
.flac 00;36
.mid 00;36
.midi 00;36
.mka 00;36
.mp3 00;36
.mpc 00;36
.ogg 00;36
.ra 00;36
.wav 00;36
# http://wiki.xiph.org/index.php/MIME_Types_and_File_Extensions
.axa 00;36
.oga 00;36
.spx 00;36
.xspf 00;36
```

Linux alias命令

Linux alias命令用于设置指令的别名。

用户可利用alias，自定指令的别名。若仅输入alias，则可列出目前所有的别名设置。alias的效力仅及于该次登入的操作。若要每次登入是即自动设好别名，可在.profile或.cshrc中设定指令的别名。

语法

```
alias[别名]=[指令名称]
```

参数说明：若不加任何参数，则列出目前所有的别名设置。

实例

给命令设置别名

```
# alias lx=ls
# lx
anaconda-ks.cfg Desktop install.log install.log.syslog qte
```

Linux clear命令

Linux clear命令用于清除屏幕。

语法

```
clear
```

实例

清屏

```
#clear
```

Linux reset命令

Linux reset命令其实和 tset 是一同个命令，它的用途是设定终端机的状态。一般而言，这个命令会自动的从环境变数、命令列或是其它的组态档决定目前终端机的型态。如果指定型态是 '?' 的话，这个程序会要求使用者输入终端机的型别。

由于这个程序会将终端机设回原始的状态，除了在 login 时使用外，当系统终端机因为程序不正常执行而进入一些奇怪的状态时，你也可以用它来重设终端机o 例如不小心把二进位档用 cat 指令进到终端机，常会有终端机不再回应键盘输入，或是回应一些奇怪字元的问题。此时就可以用 reset 将终端机回复至原始状态。

语法

```
tset [-IQqrs] [-] [-e ch] [-i ch] [-k ch] [-m mapping] [terminal]
```

参数说明：

- -p 将终端机类别显示在屏幕上，但不做设定的动作。这个命令可以用来取得目前终端机的类别。
- -e ch 将 erase 字元设成 ch
- -i ch 将中断字元设成 ch
- -k ch 将删除一行的字元设成 ch
- -I 不要做设定的动作，如果没有使用选项 -Q 的话，erase、中断及删除字元的目前值依然会送到屏幕上。
- -Q 不要显示 erase、中断及删除字元的值到屏幕上。
- -r 将终端机类别印在屏幕上。
- -s 将设定 TERM 用的命令用字串的型式送到终端机中，通常在 .login 或 .profile 中用。

实例

让使用者输入一个终端机型别并将终端机设到该型别的预设状态

```
# reset ?
```

将 erase 字元设定 control-h

```
# reset -e ^B
```

将设定用的字串显示在屏幕上

```
# reset -s
Erase is control-B (^B).
Kill is control-U (^U).
Interrupt is control-C (^C).
TERM=xterm;
```

Linux enable命令

Linux enable命令用于启动或关闭 shell 内建指令。

若要执行的文件名称与shell内建指令相同，可用enable -n来关闭shell内建指令。若不加-n参数，enable可重新启动关闭的指令。

语法

```
enable [-n][-all][内建指令]
```

参数说明：

- -n　关闭指定的shell内建指令。
- -all　显示shell所有关闭与启动的指令。

实例

显示shell内置命令

```
# enable //显示shell命令
enable .
enable :
enable [
enable alias
enable bg
enable bind
enable break
enable builtin
enable caller
enable cd
enable command
enable compgen
enable complete
enable compopt
enable continue
enable declare
enable dirs
enable disown
enable echo
enable enable
enable eval
enable exec
enable exit
enable export
enable false
enable fc
enable fg
enable getopts
enable hash
enable help
enable history
enable jobs
enable kill
enable let
enable local
enable logout
enable mapfile
enable popd
enable printf
enable pushd
enable pwd
enable read
enable readarray
enable readonly
enable return
enable set
enable shift
enable shopt
enable source
enable suspend
enable test
enable times
enable trap
enable true
enable type
enable typeset
enable ulimit
enable umask
enable unalias
enable unset
enable wait
```

Linux dmesg命令

Linux dmesg命令用于显示开机信息。

kernel会将开机信息存储在ring buffer中。您若是开机时来不及查看信息，可利用dmesg来查看。开机信息亦保存在/var/log目录中，名称为dmesg的文件里。

语法

```
dmesg [-cn][-s <缓冲区大小>]
```

参数说明：

- -c 　显示信息后，清除ring buffer中的内容。
- -s<缓冲区大小> 　预设置为8196，刚好等于ring buffer的大小。
- -n 　设置记录信息的层级。

实例

显示开机信息

```
# dmesg |less
WARNING: terminal is not fully functional
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 2.6.32-21-generic (buildd@rothera) (gcc version 4.4.3 (Ub
untu 4.4.3-4ubuntu5) ) #32-Ubuntu SMP Fri Apr 16 08:10:02 UTC 2010 (Ubuntu 2.6.32-21.3
2-generic 2.6.32.11+drm33.2)
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   NSC Geode by NSC
[    0.000000]   Cyrix CyrixInstead
[    0.000000]   Centaur CentaurHauls
[    0.000000]   Transmeta GenuineTMx86
[    0.000000]   Transmeta TransmetaCPU
[    0.000000]   UMC UMC UMC UMC
[    0.000000] BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: 0000000000000000 - 000000000009f800 (usable)
[    0.000000] BIOS-e820: 000000000009f800 - 00000000000a0000 (reserved)
[    0.000000] BIOS-e820: 00000000000ca000 - 00000000000cc000 (reserved)
[    0.000000] BIOS-e820: 00000000000dc000 - 00000000000e0000 (reserved)
[    0.000000] BIOS-e820: 00000000000e4000 - 0000000000100000 (reserved)
[    0.000000] BIOS-e820: 0000000000100000 - 000000003fef0000 (usable)
[    0.000000] BIOS-e820: 000000003fef0000 - 000000003feff000 (ACPI data)
[    0.000000] BIOS-e820: 000000003feff000 - 000000003ff00000 (ACPI NVS)

……省略部分内容
```

显示开机信息

```
#pwd   //查看当前所在目录
/home/hnlinux/
# dmesg > boot.msg //将开机信息保存到 boot.msg文件中
#ls //显示当前目录文件
boot.msg
```

Linux depmod命令

Linux depmod命令用于分析可载入模块的相依性。

depmod(depend module)可检测模块的相依性，供modprobe在安装模块时使用。

语法

```
depmod [-adeisvV][-m <文件>][--help][模块名称]
```

参数说明：

- -a或--all 　分析所有可用的模块。
- -d或debug 　执行排错模式。
- -e 　输出无法参照的符号。
- -i 　不检查符号表的版本。
- -m<文件>或system-map<文件> 　使用指定的符号表文件。
- -s或--system-log 　在系统记录中记录错误。
- -v或--verbose 　执行时显示详细的信息。
- -V或--version 　显示版本信息。
- --help 　显示帮助。

实例

显示可用模块

```
# depmod -a //显示可用模块
```

Linux declare命令

Linux declare命令用于声明 shell 变量。

declare为shell指令，在第一种语法中可用来声明变量并设置变量的属性([rix]即为变量的属性），在第二种语法中可用来显示shell函数。若不加上任何参数，则会显示全部的shell变量与函数(与执行set指令的效果相同)。

语法

```
declare [+/-][rxi][变量名称＝设置值] 或 declare -f
```

参数说明：

- +/-　"-"可用来指定变量的属性，"+"则是取消变量所设的属性。
- -f　仅显示函数。
- r　将变量设置为只读。
- x　指定的变量会成为环境变量，可供shell以外的程序来使用。
- i　[设置值]可以是数值，字符串或运算式。

实例

声明整数型变量

```
# declare -i ab //声明整数型变量
# ab=56 //改变变量内容
# echo $ab //显示变量内容
56
```

改变变量属性

```
# declare -i ef //声明整数型变量
# ef=1  //变量赋值（整数值）
# echo $ef //显示变量内容
1
# ef="wer" //变量赋值（文本值）
# echo $ef
0
# declare +i ef //取消变量属性
# ef="wer"
# echo $ef
wer
```

设置变量只读

```
# declare -r ab //设置变量为只读
# ab=88 //改变变量内容
-bash: ab: 只读变量
# echo $ab //显示变量内容
56
```

声明数组变量

```
# declare -a cd='([0]="a" [1]="b" [2]="c")' //声明数组变量
# echo ${cd[1]}
b //显示变量内容

# echo ${cd[@]} //显示整个数组变量内容
a b c
```

显示函数

```
# declare -f
command_not_found_handle ()
{
  if [ -x /usr/lib/command-not-found ]; then
    /usr/bin/python /usr/lib/command-not-found -- $1;
    return $?;
  else
    if [ -x /usr/share/command-not-found ]; then
      /usr/bin/python /usr/share/command-not-found -- $1;
      return $?;
    else
      return 127;
    fi;
  fi
}
```

Linux crontab命令

Linux crontab是用来定期执行程序的命令。

当安装完成操作系统之后，默认便会启动此任务调度命令。

crond命令每分锺会定期检查是否有要执行的工作，如果有要执行的工作便会自动执行该工作。

而linux任务调度的工作主要分为以下两类：

- 1、系统执行的工作：系统周期性所要执行的工作，如备份系统数据、清理缓存
- 2、个人执行的工作：某个用户定期要做的工作，例如每隔10分钟检查邮件服务器是否有新信，这些工作可由每个用户自行设置

语法

```
crontab [ -u user ] file
```

或

```
crontab [ -u user ] { -l | -r | -e }
```

说明：

crontab 是用来让使用者在固定时间或固定间隔执行程序之用，换句话说，也就是类似使用者的时程表。

-u user 是指设定指定 user 的时程表，这个前提是你必须要有其权限(比如说是 root)才能够指定他人的时程表。如果不使用 -u user 的话，就是表示设定自己的时程表。

参数说明：

- -e : 执行文字编辑器来设定时程表，内定的文字编辑器是 VI，如果你想用别的文字编辑器，则请先设定 VISUAL 环境变数来指定使用那个文字编辑器(比如说 setenv VISUAL joe)
- -r : 删除目前的时程表
- -l : 列出目前的时程表

时程表的格式如下：

```
f1 f2 f3 f4 f5 program
```

- 其中 f1 是表示分钟，f2 表示小时，f3 表示一个月份中的第几日，f4 表示月份，f5 表示一个星期中的第几天。program 表示要执行的程序。
- 当 f1 为 时表示每分钟都要执行 *program，f2 为* 时表示每小时都要执行程序，其馀类推
- 当 f1 为 a-b 时表示从第 a 分钟到第 b 分钟这段时间内要执行，f2 为 a-b 时表示从第 a 到第 b 小时都要执行，其馀类推
- 当 f1 为 */n* 时表示每 *n* 分钟个时间间隔执行一次，*f2* 为 /n 表示每 n 小时个时间间隔执行一次，其馀类推
- 当 f1 为 a, b, c,... 时表示第 a, b, c,... 分钟要执行，f2 为 a, b, c,... 时表示第 a, b, c...个小时要执行，其馀类推

使用者也可以将所有的设定先存放在文件中，用 crontab file 的方式来设定时程表。

实例

每月每天每小时的第 0 分钟执行一次 /bin/ls

```
0 7 * * * /bin/ls
```

在 12 月内, 每天的早上 6 点到 12 点中，每隔 20 分钟执行一次 /usr/bin/backup

```
0 6-12/3 * 12 * /usr/bin/backup
```

周一到周五每天下午 5:00 寄一封信给 alex@domain.name

```
0 17 * * 1-5 mail -s "hi" alex@domain.name < /tmp/maildata
```

每月每天的午夜 0 点 20 分, 2 点 20 分, 4 点 20 分....执行 echo "haha"

```
20 0-23/2 * * * echo "haha"
```

下面再看看几个具体的例子：

```
0 */2 * * * /sbin/service httpd restart  意思是每两个小时重启一次apache

50 7 * * * /sbin/service sshd start  意思是每天7：50开启ssh服务

50 22 * * * /sbin/service sshd stop  意思是每天22：50关闭ssh服务

0 0 1,15 * * fsck /home  每月1号和15号检查/home 磁盘

1 * * * * /home/bruce/backup  每小时的第一分执行 /home/bruce/backup这个文件

00 03 * * 1-5 find /home "*.xxx" -mtime +4 -exec rm {} \;  每周一至周五3点钟，在目录/home中，

30 6 */10 * * ls  意思是每月的1、11、21、31日是的6：30执行一次ls命令
```

注意：当程序在你所指定的时间执行后，系统会寄一封信给你，显示该程序执行的内容，若是你不希望收到这样的信，请在每一行空一格之后加上 > /dev/null 2>&1 即可

Linux clock命令

Linux clock命令用于调整 RTC 时间。

RTC 是电脑内建的硬件时间，执行这项指令可以显示现在时刻，调整硬件时钟的时间，将系统时间设成与硬件时钟之时间一致，或是把系统时间回存到硬件时钟。

语法

```
clock [--adjust][--debug][--directisa][--getepoch][--hctosys][--set --date="<日期时间>"][-
```

参数说明：

- --adjust　第一次使用"--set"或"--systohc"参数设置硬件时钟，会在/etc目录下产生一个名称为adjtime的文件。当再次使用这两个参数调整硬件时钟，此文件便会记录两次调整间之差异，日后执行clock指令加上"--adjust"参数时，程序会自动根 据记录文件的数值差异，计算出平均值，自动调整硬件时钟的时间。
- --debug　详细显示指令执行过程，便于排错或了解程序执行的情形。
- --directisa　告诉clock指令不要通过/dev/rtc设备文件，直接对硬件时钟进行存取。这个参数适用于仅有ISA总线结构的老式电脑。
- --getepoch　把系统核心内的硬件时钟新时代数值，呈现到标准输出设备。
- --hctosys　Hardware Clock to System Time，把系统时间设成和硬件时钟一致。由于这个动作将会造成系统全面更新文件的存取时间，所以最好在系统启动时就执行它。
- --set--date　设置硬件时钟的日期和时间。
- --setepoch--epoch=<年份>　设置系统核心之硬件时钟的新时代数值，年份以四位树字表示。
- --show　读取硬件时钟的时间，并将其呈现至标准输出设备。
- --systohc　System Time to Hardware Clock，将系统时间存回硬件时钟内。
- --test　仅作测试，并不真的将时间写入硬件时钟或系统时间。
- --utc　把硬件时钟上的时间时为CUT，有时也称为UTC或UCT。
- --version　显示版本信息。

实例

获取当前的时间

```
# clock //获取当前的时间
```

显示UTC时间

```
# clock -utc //显示UTC时间
```

Linux chroot命令

Linux chroot命令用于改变根目录。

chroot(change root)命令把根目录换成指定的目的目录。

、

语法

```
chroot [--help][--version][目的目录][执行指令...]
```

参数说明：

- --help 　在线帮助。
- --version 　显示版本信息。

实例

改变根目录

```
# chroot /mnt/ls //改变根目录
```

Linux insmod命令

Linux insmod(install module)命令用于载入模块。

Linux有许多功能是通过模块的方式，在需要时才载入kernel。如此可使kernel较为精简，进而提高效率，以及保有较大的弹性。这类可载入的模块，通常是设备驱动程序。

语法

```
insmod [-fkmpsvxX][-o <模块名称>][模块文件][符号名称 = 符号值]
```

参数说明：

- -f 不检查目前kernel版本与模块编译时的kernel版本是否一致，强制将模块载入。
- -k 将模块设置为自动卸除。
- -m 输出模块的载入信息。
- -o<模块名称> 指定模块的名称，可使用模块文件的文件名。
- -p 测试模块是否能正确地载入kernel。
- -s 将所有信息记录在系统记录文件中。
- -v 执行时显示详细的信息。
- -x 不要汇出模块的外部符号。
- -X 汇出模块所有的外部符号，此为预设置。

实例

加载模块

```
# insmod led.o
//向内核加载模块
```

Linux rpm命令

Linux rpm命令用于管理套件。

rmp(redhat package manager)原本是Red Hat Linux发行版专门用来管理Linux各项套件的程序，由于它遵循GPL规则且功能强大方便，因而广受欢迎。逐渐受到其他发行版的采用。RPM套件管理方式的出现，让Linux易于安装，升级，间接提升了Linux的适用度。

语法

```
rpm [-acdhilqRsv][-b<完成阶段><套间档>+][-e<套件挡>][-f<文件>+][-i<套件档>][-p<套件档>+][-U<套
```

参数说明：

- -a　查询所有套件。
- -b<完成阶段><套件档>+或-t <完成阶段><套件档>+　设置包装套件的完成阶段，并指定套件档的文件名称。
- -c　只列出组态配置文件，本参数需配合"-l"参数使用。
- -d　只列出文本文件，本参数需配合"-l"参数使用。
- -e<套件档>或--erase<套件档>　删除指定的套件。
- -f<文件>+　查询拥有指定文件的套件。
- -h或--hash　套件安装时列出标记。
- -i　显示套件的相关信息。
- -i<套件档>或--install<套件档>　安装指定的套件档。
- -l　显示套件的文件列表。
- -p<套件档>+　查询指定的RPM套件档。
- -q　使用询问模式，当遇到任何问题时，rpm指令会先询问用户。
- -R　显示套件的关联性信息。
- -s　显示文件状态，本参数需配合"-l"参数使用。
- -U<套件档>或--upgrade<套件档> 升级指定的套件档。
- -v　显示指令执行过程。
- -vv　详细显示指令执行过程，便于排错。
- -addsign<套件档>+　在指定的套件里加上新的签名认证。
- --allfiles　安装所有文件。
- --allmatches　删除符合指定的套件所包含的文件。
- --badreloc　发生错误时，重新配置文件。
- --buildroot<根目录>　设置产生套件时，欲当作根目录的目录。
- --changelog　显示套件的更改记录。

- --checksig<套件档>+　检验该套件的签名认证。
- --clean　完成套件的包装后，删除包装过程中所建立的目录。
- --dbpath<数据库目录>　设置欲存放RPM数据库的目录。
- --dump　显示每个文件的验证信息。本参数需配合"-l"参数使用。
- --excludedocs　安装套件时，不要安装文件。
- --excludepath<排除目录>　忽略在指定目录里的所有文件。
- --force　强行置换套件或文件。
- --ftpproxy<主机名称或IP地址>　指定FTP代理服务器。
- --ftpport<通信端口>　设置FTP服务器或代理服务器使用的通信端口。
- --help　在线帮助。
- --httpproxy<主机名称或IP地址>　指定HTTP代理服务器。
- --httpport<通信端口>　设置HTTP服务器或代理服务器使用的通信端口。
- --ignorearch　不验证套件档的结构正确性。
- --ignoreos　不验证套件档的结构正确性。
- --ignoresize　安装前不检查磁盘空间是否足够。
- --includedocs　安装套件时，一并安装文件。
- --initdb　确认有正确的数据库可以使用。
- --justdb　更新数据库，当不变动任何文件。
- --nobulid　不执行任何完成阶段。
- --nodeps　不验证套件档的相互关联性。
- --nofiles　不验证文件的属性。
- --nogpg　略过所有GPG的签名认证。
- --nomd5　不使用MD5编码演算确认文件的大小与正确性。
- --nopgp　略过所有PGP的签名认证。
- --noorder　不重新编排套件的安装顺序，以便满足其彼此间的关联性。
- --noscripts　不执行任何安装Script文件。
- --notriggers　不执行该套件包装内的任何Script文件。
- --oldpackage　升级成旧版本的套件。
- --percent　安装套件时显示完成度百分比。
- --pipe<执行指令>　建立管道，把输出结果转为该执行指令的输入数据。
- --prefix<目的目录>　若重新配置文件，就把文件放到指定的目录下。
- --provides　查询该套件所提供的兼容度。
- --queryformat<档头格式>　设置档头的表示方式。
- --querytags　列出可用于档头格式的标签。
- --rcfile<配置文件>　使用指定的配置文件。
- --rebulid<套件档>　安装原始代码套件，重新产生二进制文件的套件。
- --rebuliddb　以现有的数据库为主，重建一份数据库。
- --recompile<套件档>　此参数的效果和指定"--rebulid"参数类似，当不产生套件档。
- --relocate<原目录>=<新目录>　把本来会放到原目录下的文件改放到新目录。
- --replacefiles　强行置换文件。

- --replacepkgs　强行置换套件。
- --requires　查询该套件所需要的兼容度。
- --resing<套件档>+　删除现有认证，重新产生签名认证。
- --rmsource　完成套件的包装后，删除原始代码。
- --rmsource<文件>　删除原始代码和指定的文件。
- --root<根目录>　设置欲当作根目录的目录。
- --scripts　列出安装套件的Script的变量。
- --setperms　设置文件的权限。
- --setugids　设置文件的拥有者和所属群组。
- --short-circuit　直接略过指定完成阶段的步骤。
- --sign　产生PGP或GPG的签名认证。
- --target=<安装平台>+　设置产生的套件的安装平台。
- --test　仅作测试，并不真的安装套件。
- --timecheck<检查秒数>　设置检查时间的计时秒数。
- --triggeredby<套件档>　查询该套件的包装者。
- --triggers　展示套件档内的包装Script。
- --verify　此参数的效果和指定"-q"参数相同。
- --version　显示版本信息。
- --whatprovides<功能特性>　查询该套件对指定的功能特性所提供的兼容度。
- --whatrequires<功能特性>　查询该套件对指定的功能特性所需要的兼容度。

实例

安装软件

```
# rpm -hvi dejagnu-1.4.2-10.noarch.rpm
警告：dejagnu-1.4.2-10.noarch.rpm: V3 DSA 签名：NOKEY, key ID db42a60e
准备...
########################################### [100%]
```

显示软件安装信息

```
# rpm -qi dejagnu-1.4.2-10.noarch.rpm

【第1次更新 教程、类似命令关联】
```

Linux grpconv命令

Linux grpconv(group convert to shadow password)命令用于开启群组的投影密码。

Linux系统里的用户和群组密码，分别存放在/etc目录下的passwd和group文件中。因系统运作所需，任何人都得以读取它们，造成安全上的破绽。投影密码将文件内的密码改存在/etc目录下的shadow和gshadow文件内，只允许系统管理者读取，同时把原密码置换为"x"字符。投影密码的功能可随时开启或关闭，您只需执行grpconv指令就能开启群组投影密码。

语法

```
grpconv
```

Linux pwunconv命令

Linux pwunconv命令用于关闭用户的投影密码。

执行pwunconv指令可以关闭用户投影密码，它会把密码从shadow文件内，重回存到passwd文件里。

语法

```
pwunconv
```

实例

关闭用户的投影密码

```
# pwunconv
```

Linux export命令

Linux export命令用于设置或显示环境变量。

在shell中执行程序时，shell会提供一组环境变量。export可新增，修改或删除环境变量，供后续执行的程序使用。export的效力仅及于该次登陆操作。

语法

```
export [-fnp][变量名称]=[变量设置值]
```

参数说明：

- -f 　代表[变量名称]中为函数名称。
- -n 　删除指定的变量。变量实际上并未删除，只是不会输出到后续指令的执行环境中。
- -p 　列出所有的shell赋予程序的环境变量。

实例

列出当前所有的环境变量

```
# export -p //列出当前的环境变量值
declare -x HOME="/root"
declare -x LANG="zh_CN.UTF-8"
declare -x LANGUAGE="zh_CN:zh"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="root"
declare -x LS_COLORS=""
declare -x MAIL="/var/mail/root"
declare -x OLDPWD
declare -x PATH="/opt/toolchains/arm920t-eabi/bin:/opt/toolchains/arm920t-eabi/bin:/usr/l
declare -x PWD="/root"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SPEECHD_PORT="6560"
declare -x SSH_CLIENT="192.168.1.65 1674 22"
declare -x SSH_CONNECTION="192.168.1.65 1674 192.168.1.3 22"
declare -x SSH_TTY="/dev/pts/2"
declare -x TERM="XTERM"
declare -x USER="root"
declare -x XDG_SESSION_COOKIE="93b5d3d03e032c0cf892a4474bebda9f-1273864738.954257-3402064
```

定义环境变量

```
# export MYENV //定义环境变量
# export -p //列出当前的环境变量
declare -x HOME="/root"
declare -x LANG="zh_CN.UTF-8"
declare -x LANGUAGE="zh_CN:zh"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="root"
declare -x LS_COLORS=""
declare -x MAIL="/var/mail/root"
declare -x MYENV
declare -x OLDPWD
declare -x PATH="/opt/toolchains/arm920t-eabi/bin:/opt/toolchains/arm920t-eabi/bin:/usr/l
declare -x PWD="/root"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SPEECHD_PORT="6560"
declare -x SSH_CLIENT="192.168.1.65 1674 22"
declare -x SSH_CONNECTION="192.168.1.65 1674 192.168.1.3 22"
declare -x SSH_TTY="/dev/pts/2"
declare -x TERM="XTERM"
declare -x USER="root"
declare -x XDG_SESSION_COOKIE="93b5d3d03e032c0cf892a4474bebda9f-1273864738.954257-3402064
```

定义环境变量赋值

```
# export MYENV=7 //定义环境变量并赋值
# export -p
declare -x HOME="/root"
declare -x LANG="zh_CN.UTF-8"
declare -x LANGUAGE="zh_CN:zh"
declare -x LESSCLOSE="/usr/bin/lesspipe %s %s"
declare -x LESSOPEN="| /usr/bin/lesspipe %s"
declare -x LOGNAME="root"
declare -x LS_COLORS=""
declare -x MAIL="/var/mail/root"
declare -x MYENV="7"
declare -x OLDPWD
declare -x PATH="/opt/toolchains/arm920t-eabi/bin:/opt/toolchains/arm920t-eabi/bin:/usr/l
declare -x PWD="/root"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SPEECHD_PORT="6560"
declare -x SSH_CLIENT="192.168.1.65 1674 22"
declare -x SSH_CONNECTION="192.168.1.65 1674 192.168.1.3 22"
declare -x SSH_TTY="/dev/pts/2"
declare -x TERM="XTERM"
declare -x USER="root"
declare -x XDG_SESSION_COOKIE="93b5d3d03e032c0cf892a4474bebda9f-1273864738.954257-3402064
```

Linux eval命令

Linux eval命令用于重新运算求出参数的内容。

eval可读取一连串的参数，然后再依参数本身的特性来执行。

语法

```
eval [参数]
```

参数说明：参数不限数目，彼此之间用分号分开。

实例

连接多个命令

```
# eval enable;ls //连接多个命令
enable .
enable :
enable [
enable alias
enable bg
enable bind
enable break
enable builtin
enable caller
enable cd
enable command
enable compgen
enable complete
enable compopt
enable continue
enable declare
enable dirs
enable disown
enable echo
enable enable
enable eval
enable exec
enable exit
enable export
enable false
enable fc
enable fg
enable getopts
enable hash
enable help
enable history
enable jobs
enable kill
enable let
enable local
enable logout
enable mapfile
enable popd
enable printf
enable pushd
enable pwd
enable read
enable readarray
enable readonly
enable return
enable set
enable shift
enable shopt
enable source
enable suspend
enable test
enable times
enable trap
enable true
enable type
enable typeset
enable ulimit
enable umask
enable unalias
enable unset
enable wait
```

Linux set命令

Linux set命令用于设置shell。

set指令能设置所使用shell的执行方式，可依照不同的需求来做设置。

语法

```
set [+-abCdefhHklmnpPtuvx]
```

参数说明：

- -a 　标示已修改的变量，以供输出至环境变量。
- -b 　使被中止的后台程序立刻回报执行状态。
- -C 　转向所产生的文件无法覆盖已存在的文件。
- -d 　Shell预设会用杂凑表记忆使用过的指令，以加速指令的执行。使用-d参数可取消。
- -e 　若指令传回值不等于0，则立即退出shell。
- -f　 　取消使用通配符。
- -h 　自动记录函数的所在位置。
- -H Shell 　可利用"!"加<指令编号>的方式来执行history中记录的指令。
- -k 　指令所给的参数都会被视为此指令的环境变量。
- -l 　记录for循环的变量名称。
- -m 　使用监视模式。
- -n 　只读取指令，而不实际执行。
- -p 　启动优先顺序模式。
- -P 　启动-P参数后，执行指令时，会以实际的文件或目录来取代符号连接。
- -t 　执行完随后的指令，即退出shell。
- -u 　当执行时使用到未定义过的变量，则显示错误信息。
- -v 　显示shell所读取的输入值。
- -x 　执行指令后，会先显示该指令及所下的参数。
- +<参数> 　取消某个set曾启动的参数。

实例

显示环境变量

```
# set
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="00" [2]="15" [3]="1" [4]="release" [5]="i386-redhat-linux-gnu
BASH_VERSION='3.00.15(1)-release'
COLORS=/etc/DIR_COLORS.xterm
COLUMNS=99
DIRSTACK=()
EUID=0
GROUPS=()
G_BROKEN_FILENAMES=1
HISTFILE=/root/.bash_history
HISTFILESIZE=1000
HISTSIZE=1000
HOME=/root
HOSTNAME=hnlinux
HOSTTYPE=i386
IFS=/plinux> '
INPUTRC=/etc/inputrc
KDEDIR=/usr
LANG=zh_CN.GB2312
LESSOPEN='|/usr/bin/lesspipe.sh %s'
LINES=34
L
MAIL=/var/spool/mail/root
MAILCHECK=60
OLDPWD=/home/uptech
OPTERR=1
OPTIND=1
OSTYPE=linux-gnu
PATH=/usr/kerberos/sbin:/usr/kerberos/bin:/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/
PIPESTATUS=([0]="2")
PPID=26005
PROMPT_COMMAND='echo -ne "
```

Linux minfo命令

Linux minfo命令用于显示MS-DOS文件系统的各项参数。

minfo为mtools工具指令，可显示MS-DOS系统磁盘的各项参数，包括磁区数，磁头数...等。

语法

```
</>minfo [-v][驱动器代号]
```

参数说明：

- -v 除了一般信息外，并显示可开机磁区的内容。

实例

显示DOS系统参数

```
# minfo -v C: //显示系统参数
```

Linux lsmod命令

Linux lsmod命令用于显示已载入系统的模块。

执行lsmod(list modules)指令，会列出所有已载入系统的模块。Linux操作系统的核心具有模块化的特性，应此在编译核心时，务须把全部的功能都放入核心。您可以将这些功能编译成一个个单独的模块，待需要时再分别载入。

语法

```
lsmod
```

实例

显示模块信息

```
# lsmod
Module         Size Used by
nfsd          238935 11
lockd          64849 1 nfsd
nfs_acl         2245 1 nfsd
auth_rpcgss      33735 1 nfsd
sunrpc         193181 10 nfsd,lockd,nfs_acl,auth_rpcgss
exportfs        3437 1 nfsd
xt_TCPMSS        2931 1
xt_tcpmss        1197 1
xt_tcpudp        2011 1
iptable_mangle     2771 1
ip_tables        9991 1 iptable_mangle
x_tables        14299 4 xt_TCPMSS,xt_tcpmss,xt_tcpudp,ip_tables
pppoe          8943 2
pppox          2074 1 pppoe
binfmt_misc       6587 1
snd_ens1371      18814 0
gameport        9089 1 snd_ens1371
snd_ac97_codec    100646 1 snd_ens1371
ac97_bus        1002 1 snd_ac97_codec
snd_pcm_oss      35308 0
snd_mixer_oss     13746 1 snd_pcm_oss
snd_pcm        70662 3 snd_ens1371,snd_ac97_codec,snd_pcm_oss
snd_seq_dummy      1338 0
snd_seq_oss      26726 0
snd_seq_midi      4557 0
snd_rawmidi      19056 2 snd_ens1371,snd_seq_midi
snd_seq_midi_event   6003 2 snd_seq_oss,snd_seq_midi
snd_seq        47263 6 snd_seq_dummy,snd_seq_oss,snd_seq_midi,snd_seq_midi_event
snd_timer       19098 2 snd_pcm,snd_seq
snd_seq_device     5700 5 snd_seq_dummy,snd_seq_oss,snd_seq_midi,snd_rawmidi,snd_seq
fbcon         35102 71
tileblit        2031 1 fbcon
font          7557 1 fbcon
bitblit         4707 1 fbcon
ppdev          5259 0
softcursor       1189 1 bitblit
snd          54148 10 snd_ens1371,snd_ac97_codec,snd_pcm_oss,snd_mixer_oss,snd_pcm,snd_se
psmouse        63245 0
serio_raw        3978 0
soundcore        6620 1 snd
parport_pc       25962 1
snd_page_alloc     7076 1 snd_pcm
vga16fb        11385 1
intel_agp       24177 1
vgastate        8961 1 vga16fb
i2c_piix4        8335 0
shpchp         28820 0
agpgart        31724 1 intel_agp
lp          7028 0
parport        32635 3 ppdev,parport_pc,lp
mptspi         14652 2
mptscsih        31325 1 mptspi
pcnet32        28890 0
floppy         53016 0
mii          4381 1 pcnet32
mptbase        83022 2 mptspi,mptscsih
scsi_transport_spi   21096 1 mptspi
```

Linux liloconfig命令

Linux liloconfig命令用于设置核心载入，开机管理程序。

liloconfig是Slackware发行版专门用来调整lilo设置的程序。它通过互动式操作界面，让用户能够利用键盘上的方向键等，轻易地操控lilo的安装，设置作业，而无须下达各种参数或撰写配置文件。

语法

```
liloconfig
```

实例

执行liloconfig命令

```
# liloconfig
```

Linux lilo命令

Linux lilo命令用于安装核心载入，开机管理程序。

lilo(linux loader)是个Linux系统核心载入程序，同时具备管理开机的功能。单独执行lilo指令，它会读取/etc/目录下的lilo.conf配置文件，然后根据其内容安装lilo。

语法

```
lilo [-clqtV][-b<外围设备代号>][-C<配置文件>][-d<延迟时间>][－D<识别标签>][-f<几何参数文件>][-i<开
```

参数说明：

- -b<外围设备代号>　指定安装lilo之处的外围设备代号。
- -c　使用紧致映射模式。
- -C<配置文件>　指定lilo的配置文件。
- -d<延迟时间>　设置开机延迟时间。
- -D<识别标签>　指定开机后预设启动的操作系统，或系统核心识别标签。
- -f<几何参数文件>　指定磁盘的几何参数配置文件。
- -i<开机磁区文件>　指定欲使用的开机磁区文件，预设是/boot目录里的boot.b文件。
- -I<识别标签>　显示系统核心存放之处。
- -l　产生线形磁区地址。
- -m<映射文件>　指定映射文件。
- -P<fix/ignore>　决定要修复或忽略分区表的错误。
- -q　列出映射的系统核心文件。
- -r<根目录>　设置系统启动时欲挂入成为根目录的目录。
- -R<执行指令>　设置下次启动系统时，首先执行的指令。
- -s<备份文件>　指定备份文件。
- -S<备份文件>　强制指定备份文件。
- -t　不执行指令，仅列出实际执行会进行的动作。
- -u<外围色设备代号>　删除lilo。
- -U<外围设备代号>　此参数的效果和指定"-u"参数类似，当不检查时间戳记。
- -v　显示指令执行过程。
- -V　显示版本信息。

实例

安装lilo到第一台SCSI硬盘的第三个主要分区，采用3级模式。

```
# lilo -b /dev/sda3 -v -v -v
```

指定安装lilo的配置文件和备份文件。

```
# lilo -C /etc/lilo.conf2 -s /boot/boot. Backup
```

Linux kbdconfig命令

Linux kbdconfig命令用于设置键盘类型。

kbdconfig(Red Hat Linux才有的指令)是一个用来设置键盘的程序，提供图形化的操作界面。kbdconfig实际上是修改/etc/sysconfig/keyboard的键盘配置文件。

语法

```
kbdconfig [--back][--test]
```

参数：

- --back 执行时将预设的Cancel按钮更改为Back按钮。
- --test 仅作测试，不会实际更改设置。

实例

键盘设置：

```
# kdbconfig //设置键盘
```

Linux modprobe命令

Linux modprobe命令用于自动处理可载入模块。

modprobe可载入指定的个别模块，或是载入一组相依的模块。modprobe会根据depmod所产生的相依关系，决定要载入哪些模块。若在载入过程中发生错误，在modprobe会卸载整组的模块。

语法

```
modprobe [-acdlrtvV][--help][模块文件][符号名称 = 符号值]
```

参数：

- -a或--all 载入全部的模块。
- -c或--show-conf 显示所有模块的设置信息。
- -d或--debug 使用排错模式。
- -l或--list 显示可用的模块。
- -r或--remove 模块闲置不用时，即自动卸载模块。
- -t或--type 指定模块类型。
- -v或--verbose 执行时显示详细的信息。
- -V或--version 显示版本信息。
- -help 显示帮助。

实例

安装软驱模块：

```
[root@w3cschool.cc ~]# modprobe -v floppy
```

卸载软驱模块：

```
[root@w3cschool.cc ~]# modprobe -v -r floppy
```

Linux ntsysv命令

Linux ntsysv命令用于设置系统的各种服务。

这是Red Hat公司遵循GPL规则所开发的程序，它具有互动式操作界面，您可以轻易地利用方向键和空格键等，开启，关闭操作系统在每个执行等级中，所要执行的系统服务。

语法

```
ntsysv [--back][--level <等级代号>]
```

参数：

- --back 在互动式界面里，显示Back钮，而非Cancel钮。
- --level <等级代号> 在指定的执行等级中，决定要开启或关闭哪些系统服务。

Linux mouseconfig命令

Linux mouseconfig命令用于设置鼠标相关参数。

mouseconfig为鼠标设置程序，可自动设置相关参数，或者用户也可以利用所提供互动模式自行设置鼠标。mouseconfig是Red Hat Linux才有的命令。

语法

```
mouseconfig [--back][--emulthree][--help][--expert][--kickstart][--noprobe][--test][--dev
```

参数：

- --back　在设置画面上显示Back按钮，而取代预设的Cancel按钮。
- --device<连接端口>　指定硬件连接端口。可用的选项有ttyS0，ttyS1，ttyS2，ttyS3与orpsaux。
- --emulthree　将二钮鼠标模拟成三钮鼠标。
- --help　显示帮助以及所有支持的鼠标类型。
- --expert　程序预设可自动判断部分设置值。若要自行设置，请使用--expert参数。
- --kickstart　让程序自动检测并保存所有的鼠标设置。
- --noprobe　不要检测鼠标设备。
- --test　测试模式，不会改变任何设置。

实例

以交互模式配置鼠标：

```
# mouseconfig -text
```

Linux passwd命令

Linux passwd命令用来更改使用者的密码

语法

```
passwd [-k] [-l] [-u [-f]] [-d] [-S] [username]
```

必要参数：

- -d 删除密码
- -f 强制执行
- -k 更新只能发送在过期之后
- -l 停止账号使用
- -S 显示密码信息
- -u 启用已被停止的账户
- -x 设置密码的有效期
- -g 修改群组密码
- -i 过期后停止用户账号

选择参数：

- --help 显示帮助信息
- --version 显示版本信息

实例

修改用户密码

```
# passwd w3cschool  //设置w3cschool用户的密码
Enter new UNIX password:  //输入新密码，输入的密码无回显
Retype new UNIX password:  //确认密码
passwd: password updated successfully
#
```

显示账号密码信息

```
# passwd -S w3cschool
w3cschool P 05/13/2010 0 99999 7 -1
```

删除用户密码

```
# passwd -d lx138
passwd: password expiry information changed.
```

Linux pwconv命令

Linux pwconv命令用于开启用户的投影密码。

Linux系统里的用户和群组密码，分别存放在名称为passwd和group的文件中， 这两个文件位于/etc目录下。因系统运作所需，任何人都得以读取它们，造成安全上的破绽。投影密码将文件内的密码改存在/etc目录下的shadow和gshadow文件内，只允许系统管理者读取，同时把原密码置换为"x"字符，有效的强化了系统的安全性。

语法

```
pwconv
```

实例

开启用户的投影密码

```
# pwconv
```

Linux rdate命令

Linux rdate命令用于显示其他主机的日期与时间。

执行rdate指令，向其他主机询问系统时间并显示出来。

语法

```
rdate [-ps][主机名称或IP地址...]
```

参数：

- -p 　显示远端主机的日期与时间。
- -s 　把从远端主机收到的日期和时间，回存到本地主机的系统时间。
- -u 传输协议使用UDP协议
- -l 使用syslog显示错误信息
- -t<时间> 设置超时时间

Linux resize命令

Linux resize命令设置终端机视窗的大小。

执行resize指令可设置虚拟终端机的视窗大小。

语法

```
resize [-cu][-s <列数> <行数>]
```

参数：

- -c 　就算用户环境并非C Shell，也用C Shell指令改变视窗大小。
- -s <列数> <行数> 　设置终端机视窗的垂直高度和水平宽度。
- -u 　就算用户环境并非Bourne Shell，也用Bourne Shell指令改变视窗大小。

实例

使用 C shell

```
[root@linux w3cschool.cc]# resize -c
set noglob;
setenv COLUMNS '99';
setenv LINES '34';
unset noglob;
```

使用 Bourne shell

```
[root@hnlinux w3cschool.cc]# resize -u
COLUMNS=99;
LINES=34;
export COLUMNS LINES;
```

设置指定大小

```
[root@hnlinux w3cschool.cc]# resize -s 80 160
```

Linux rmmod命令

Linux rmmod命令用于删除模块。

执行rmmod指令，可删除不需要的模块。Linux操作系统的核心具有模块化的特性，应此在编译核心时，务须把全部的功能都放如核心。你可以将这些功能编译成一个个单独的模块，待有需要时再分别载入它们。

语法

```
rmmod [-as][模块名称...]
```

参数：

- -a　删除所有目前不需要的模块。
- -s　把信息输出至syslog常驻服务，而非终端机界面。

实例

显示已安装的模块

```
# lsmod
Module         Size Used by
cramfs         39042 1
nfsd          238935 11
lockd          64849 1 nfsd
nfs_acl         2245 1 nfsd
auth_rpcgss      33735 1 nfsd
sunrpc         193181 10 nfsd,lockd,nfs_acl,auth_rpcgss
exportfs        3437 1 nfsd
xt_TCPMSS        2931 0
xt_tcpmss        1197 0
xt_tcpudp        2011 0
iptable_mangle     2771 0
ip_tables        9991 1 iptable_mangle
x_tables        14299 4

……省略部分结果
pppoe          8943 0
pppox          2074 1 pppoe
binfmt_misc       6587 1
snd_ens1371       18814 0
gameport        9089 1 snd_ens1371
snd_ac97_codec    100646 1 snd_ens1371
ac97_bus        1002 1 snd_ac97_codec
snd_pcm_oss      35308 0
```

卸载模块

```
# rmmod -v pppoe //卸载模块pppoe
Checking ppoe for persistent data
```

安装模块

```
# insmod -v pppoe >1.log //安装模块

~# tail -b 30 1.log //显示文件信息
```

Linux grpunconv命令

Linux grpunconv命令用于关闭群组的投影密码。

执行grpunconv指令可关闭群组投影密码，它会把密码从gshadow文件内，回存到group文件里。

语法

```
grpunconv
```

实例

未关闭的情况

```
cat /etc/gshadow | grep cdy
cdy:123456::
```

关闭影子密码

```
cat /etc/gshadow
cat: /etc/gshadow: 没有那个文件或目录
```

查看密码已经复制到 /etc/group 中了。

```
cat /etc/group | grep cdy
cdy:123456:1000:
```

Linux modinfo命令

Linux modinfo命令用于显示kernel模块的信息。

modinfo会显示kernel模块的对象文件，以显示该模块的相关信息。

语法

```
modinfo [-adhpV][模块文件]
```

参数：

- -a或--author 显示模块开发人员。
- -d或--description 显示模块的说明。
- -h或--help 显示modinfo的参数使用方法。
- -p或--parameters 显示模块所支持的参数。
- -V或--version 显示版本信息。

实例

显示sg模块的信息。

```
# modinfo sg
filename:    /lib/modules/2.6.9-42.ELsmp/kernel/drivers/scsi/sg.ko
author:     Douglas Gilbert
description:  SCSI generic (sg) driver
license:    GPL
version:    3.5.31 B0B0CB1BB59F0669A1F0D6B
parm:      def_reserved_size:size of buffer reserved for each fd
parm:      allow_dio:allow direct I/O (default: 0 (disallow))
alias:     char-major-21-*
vermagic:    2.6.9-42.ELsmp SMP 686 REGPARM 4KSTACKS gcc-3.4
depends:    scsi_mod
```

Linux time命令

Linux time命令的用途，在于量测特定指令执行时所需消耗的时间及系统资源等资讯。

例如 CPU 时间、记忆体、输入输出等等。需要特别注意的是，部分资讯在 Linux 上显示不出来。这是因为在 Linux 上部分资源的分配函式与 time 指令所预设的方式并不相同，以致于 time 指令无法取得这些资料。

语法

```
time [options] COMMAND [arguments]
```

参数：

- -o 或 --output=FILE：设定结果输出档。这个选项会将 time 的输出写入 所指定的档案中。如果档案已经存在，系统将覆写其内容。
- -a 或 --append：配合 -o 使用，会将结果写到档案的末端，而不会覆盖掉原来的内容。
- -f FORMAT 或 --format=FORMAT：以 FORMAT 字串设定显示方式。当这个选项没有被设定的时候，会用系统预设的格式。不过你可以用环境变数 time 来设定这个格式，如此一来就不必每次登入系统都要设定一次。

time 指令可以显示的资源有四大项，分别是：

- Time resources
- Memory resources
- IO resources
- Command info

详细的内容如下：

1、Time Resources

E 执行指令所花费的时间，格式是：[hour]:minute:second。请注意这个数字并不代表实际的 CPU 时间。

e 执行指令所花费的时间，单位是秒。请注意这个数字并不代表实际的 CPU 时间。

S 指令执行时在核心模式（kernel mode）所花费的时间，单位是秒。

U 指令执行时在使用者模式（user mode）所花费的时间，单位是秒。

P 执行指令时 CPU 的占用比例。其实这个数字就是核心模式加上使用者模式的 CPU 时间除以总时间。

2、Memory Resources

M 执行时所占用的实体记忆体的最大值。单位是 KB

t 执行时所占用的实体记忆体的平均值，单位是 KB

K 执行程序所占用的记忆体总量（stack+data+text）的平均大小，单位是 KB

D 执行程序的自有资料区（unshared data area）的平均大小，单位是 KB

p 执行程序的自有堆叠（unshared stack）的平均大小，单位是 KB

X 执行程序间共享内容（shared text）的平均值，单位是 KB

Z 系统记忆体页的大小，单位是 byte。对同一个系统来说这是个常数

3、IO Resources

F 此程序的主要记忆体页错误发生次数。所谓的主要记忆体页错误是指某一记忆体页已经置换到置换档（swap file)中，而且已经分配给其他程序。此时该页的内容必须从置换档里再读出来。

R 此程序的次要记忆体页错误发生次数。所谓的次要记忆体页错误是指某一记忆体页虽然已经置换到置换档中，但尚未分配给其他程序。此时该页的内容并未被破坏，不必从置换档里读出来

W 此程序被交换到置换档的次数

c 此程序被强迫中断（像是分配到的 CPU 时间耗尽）的次数

w 此程序自愿中断（像是在等待某一个 I/O 执行完毕，像是磁碟读取等等）的次数

I 此程序所输入的档案数

O 此程序所输出的档案数

r 此程序所收到的 Socket Message

s 此程序所送出的 Socket Message

k 此程序所收到的信号 (Signal)数量

4、Command Info

C 执行时的参数以及指令名称

x 指令的结束代码 (Exit Status)

-p or --portability：这个选项会自动把显示格式设定成为：

real %e user %Usys %S：这么做的目的是为了与 POSIX 规格相容。

-v or --verbose：这个选项会把所有程序中用到的资源通通列出来，不但如一般英文语句，还有说明。对不想花时间去熟习格式设定或是刚刚开始接触这个指令的人相当有用。

实例

```
1\. # time date
2\. Sun Mar 26 22:45:34 GMT-8 2006
3\.
4\. real    0m0.136s
5\. user    0m0.010s
6\. sys     0m0.070s
7\. #
```

在以上实例中，执行命令"time date"(见第1行)。

系统先执行命令"date"，第2行为命令"date"的执行结果。

第3-6行为执行命令"date"的时间统计结果，其中第4行"real"为实际时间，第5行"user"为用户CPU时间，第6行"sys"为系统CPU时间。

以上三种时间的显示格式均为MMmNN[.FFF]s。

利用下面的指令

```
time -v ps -aux
```

我们可以获得执行 ps -aux 的结果和所花费的系统资源。如下面所列的资料：

```
USER PID %CPU %MEM VSZ RSS TTY STAT START TIME COMMAND
root 1 0.0 0.4 1096 472 ? S Apr19 0:04 init
root 2 0.0 0.0 0 0 ? SW Apr19 0:00 [kflushd]
root 3 0.0 0.0 0 0 ? SW Apr19 0:00 [kpiod]
......
root 24269 0.0 1.0 2692 996 pts/3 R 12:16 0:00 ps -aux
Command being timed: "ps -aux"
User time (seconds): 0.05
System time (seconds): 0.06
Percent of CPU this job got: 68%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.16
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 0
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 238
Minor (reclaiming a frame) page faults: 46
Voluntary context switches: 0
Involuntary context switches: 0
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

Linux setup命令

Linux setup命令设置公用程序，是一个启动图形设置系统的命令。

setup 命令：用来配置X，打印设置，时区设置，系统服务，网络配置，配置，防火墙配置，验证配置，鼠标配置。

语法

```
setup
```

setup是一个设置公用程序，提供图形界面的操作方式。在setup中可设置7类的选项：

- 1.登陆认证方式
- 2.键盘组态设置
- 3.鼠标组态设置
- 4.开机时所要启动的系统服务
- 5.声卡组态设置
- 6.时区设置
- 7.X Windows组态设置

Linux sndconfig命令

Linux sndconfig命令用于设置声卡。

sndconfig为声卡设置程序，支持PnP设置，可自动检测并设置PnP声卡。

语法

```
sndconfig [--help][--noautoconfig][--noprobe]
```

参数：

- --help 显示帮助。
- --noautoconfig 不自动设置PnP的声卡。
- --noprobe 不自动检测PnP声卡。

Linux setenv命令

Linux setenv命令用于查询或显示环境变量。

setenv为tsch中查询或设置环境变量的指令。

语法

```
setenv [变量名称][变量值]
```

实例

显示环境变量

```
setenv
```

设置环境变量

```
# setenv USER lx138
```

Linux chkconfig命令

Linux chkconfig命令用于检查，设置系统的各种服务。

这是Red Hat公司遵循GPL规则所开发的程序，它可查询操作系统在每一个执行等级中会执行哪些系统服务，其中包括各类常驻服务。

语法

```
chkconfig [--add][--del][--list][系统服务] 或 chkconfig [--level <等级代号>][系统服务][on/off
```

参数：

- --add　增加所指定的系统服务，让chkconfig指令得以管理它，并同时在系统启动的叙述文件内增加相关数据。
- --del　删除所指定的系统服务，不再由chkconfig指令管理，并同时在系统启动的叙述文件内删除相关数据。
- --level<等级代号>　指定读系统服务要在哪一个执行等级中开启或关毕。

实例

列出chkconfig所知道的所有命令。

```
# chkconfig -list
```

开启服务。

```
# chkconfig telnet on //开启Telnet服务
# chkconfig -list //列出chkconfig所知道的所有的服务的情况
```

关闭服务

```
# chkconfig telnet off  //关闭Telnet服务
# chkconfig -list //列出chkconfig所知道的所有的服务的情况
```

Linux unset命令

Linux unset命令用于删除变量或函数。

unset为shell内建指令，可删除变量或函数。

语法

```
unset [-fv][变量或函数名称]
```

参数：

- -f 　仅删除函数。
- -v 　仅删除变量。

实例

删除环境变量

```
[root@w3cschool.cc ~]# lx="ls -lh" //设定环境变量
[root@w3cschool.cc ~]# $lx //使用环境变量
总用量 116K
-rw-r--r-- 1 root root 2.1K 2008-03-30 anaconda-ks.cfg
drwx------ 3 root root 4.0K 3月 30 21:22 Desktop
-rw-r--r-- 1 root root 50K 2008-03-30 install.log
-rw-r--r-- 1 root root 32K 2008-03-30 install.log.syslog
lrwxrwxrwx 1 root root  9 2008-03-30 qte -> /opt/qte/
[root@w3cschool.cc ~]# set //查看当前的环境变量
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
……省略部分内容
PROMPT_COMMAND='echo -ne "33]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}07"'
PS1='[u@h W]$ '
PS2='> '
PS4='+ '
PWD=/root
QTDIR=/usr/lib/qt-3.3
SHELL=/bin/bash
SSH_TTY=/dev/pts/4
SUPPORTED=zh_CN.UTF-8:zh_CN:zh:en_US.UTF-8:en_US:en
SYSFONT=latarcyrheb-sun16
TERM=xterm
UID=0
USER=root
_=-lh
lx='ls -lh'
[root@w3cschool.cc ~]# unset lx //删除环境变量
[root@w3cschool.cc ~]# set //显示当前环境变量
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
……省略部分内容
PROMPT_COMMAND='echo -ne "33]0;${USER}@${HOSTNAME%%.*}:${PWD/#$HOME/~}07"'
PS1='[u@h W]$ '
PS2='> '
PS4='+ '
PWD=/root
QTDIR=/usr/lib/qt-3.3
SHELL=/bin/bash
SSH_TTY=/dev/pts/4
SUPPORTED=zh_CN.UTF-8:zh_CN:zh:en_US.UTF-8:en_US:en
SYSFONT=latarcyrheb-sun16
TERM=xterm
UID=0
USER=root
_=-lh
```

Linux ulimit命令

Linux ulimit命令用于控制shell程序的资源。

ulimit为shell内建指令，可用来控制shell执行程序的资源。

语法

```
ulimit [-aHS][-c <core文件上限>][-d <数据节区大小>][-f <文件大小>][-m <内存大小>][-n <文件数目>]
```

参数：

- -a 显示目前资源限制的设定。
- -c <core文件上限> 设定core文件的最大值，单位为区块。
- -d <数据节区大小> 程序数据节区的最大值，单位为KB。
- -f <文件大小> shell所能建立的最大文件，单位为区块。
- -H 设定资源的硬性限制，也就是管理员所设下的限制。
- -m <内存大小> 指定可使用内存的上限，单位为KB。
- -n <文件数目> 指定同一时间最多可开启的文件数。
- -p <缓冲区大小> 指定管道缓冲区的大小，单位512字节。
- -s <堆叠大小> 指定堆叠的上限，单位为KB。
- -S 设定资源的弹性限制。
- -t <CPU时间> 指定CPU使用时间的上限，单位为秒。
- -u <程序数目> 用户最多可开启的程序数目。
- -v <虚拟内存大小> 指定可使用的虚拟内存上限，单位为KB。

实例

显示系统资源的设置

```
[root@w3cschool.cc ~]# ulimit -a
core file size     (blocks, -c) 0
data seg size      (kbytes, -d) unlimited
file size        (blocks, -f) unlimited
pending signals         (-i) 1024
max locked memory    (kbytes, -l) 32
max memory size     (kbytes, -m) unlimited
open files           (-n) 1024
pipe size      (512 bytes, -p) 8
POSIX message queues   (bytes, -q) 819200
stack size       (kbytes, -s) 10240
cpu time        (seconds, -t) unlimited
max user processes       (-u) 4096
virtual memory     (kbytes, -v) unlimited
file locks           (-x) unlimited
[root@w3cschool.cc ~]#
```

设置单一用户程序数目上限

```
[root@w3cschool.cc ~]# ulimit -u 500 //设置单一用户程序上限
[root@w3cschool.cc ~]# ulimit -a
core file size     (blocks, -c) 0
data seg size      (kbytes, -d) unlimited
file size        (blocks, -f) unlimited
pending signals         (-i) 1024
max locked memory    (kbytes, -l) 32
max memory size     (kbytes, -m) unlimited
open files           (-n) 1024
pipe size      (512 bytes, -p) 8
POSIX message queues   (bytes, -q) 819200
stack size       (kbytes, -s) 10240
cpu time        (seconds, -t) unlimited
max user processes       (-u) 500
virtual memory     (kbytes, -v) unlimited
file locks           (-x) unlimited
[root@w3cschool.cc ~]#
```

Linux timeconfig命令

Linux timeconfig命令用于设置时区。

这是Red Hat公司遵循GPL规则所开发的程序，它具有互动式操作界面，您可以轻易地利用方向键和空格键等，设置系统时间所属的时区。

语法

```
timeconfig [--arc][--back][--test][--utc][时区名称]
```

参数：

- --arc 　使用Alpha硬件结构的格式存储系统时间。
- --back 　在互动式界面里，显示Back钮而非Cancel钮。
- --test 　仅作测试，并不真的改变系统的时区。
- --utc 　把硬件时钟上的时间视为CUT，有时也称为UTC或UCT。

实例

```
# timeconfig //设置时区
```

Linux setconsole命令

Linux setconsole命令用于设置系统终端。

setconsole可用来指定系统终端。

语法

```
setconsole [serial][ttya][ttyb]
```

参数：

- serial 使用PROM终端。
- ttya,cua0或ttyS0 使用第 1 个串口设备作为终端。
- ttyb,cua1或ttyS1 使用第 2 个串口设备作为终端。
- video 使用主机上的现卡作为终端。

实例

设置终端

```
# setconsole ttyS0
```

Linux mkkickstart命令

Linux mkkickstart命令用于建立安装的组态文件。

mkkickstart可根据目前系统的设置来建立组态文件，供其他电脑在安装时使用。组态文件的内容包括使用语言，网络环境，系统磁盘状态，以及X Windows的设置等信息。

语法

```
mkkickstart [--bootp][--dhcp][--nonet][--nox][--version][--nfs <远端电脑:路径>]
```

参数：

- --bootp 安装与开机时，使用BOOTP。
- --dhcp 安装与开机时，使用DHCP。
- --nfs<远端电脑:路径> 使用指定的网络路径安装。
- --nonet 不要进行网络设置，即假设在没有网络环境的状态下。
- --nox 不要进行X Windows的环境设置。
- --version 显示版本信息。

实例

构建一个安装组态文件：

```
# mkkickstart --nonet -bootp
```

Linux hwclock命令

Linux hwclock命令用于显示与设定硬件时钟。

在Linux中有硬件时钟与系统时钟等两种时钟。硬件时钟是指主机板上的时钟设备，也就是通常可在BIOS画面设定的时钟。系统时钟则是指kernel中的时钟。当Linux启动时，系统时钟会去读取硬件时钟的设定，之后系统时钟即独立运作。所有Linux相关指令与函数都是读取系统时钟的设定。

语法

```
hwclock [--adjust][--debug][--directisa][--hctosys][--show][--systohc][--test]
[--utc][--version][--set --date=<日期与时间>]
```

参数：

- --adjust　hwclock每次更改硬件时钟时，都会记录在/etc/adjtime文件中。使用--adjust参数，可使hwclock根据先前的记录来估算硬件时钟的偏差，并用来校正目前的硬件时钟。
- --debug　显示hwclock执行时详细的信息。
- --directisa　hwclock预设从/dev/rtc设备来存取硬件时钟。若无法存取时，可用此参数直接以I/O指令来存取硬件时钟。
- --hctosys　将系统时钟调整为与目前的硬件时钟一致。
- --set --date=<日期与时间>　设定硬件时钟。
- --show　显示硬件时钟的时间与日期。
- --systohc　将硬件时钟调整为与目前的系统时钟一致。
- --test　仅测试程序，而不会实际更改硬件时钟。
- --utc　若要使用格林威治时间，请加入此参数，hwclock会执行转换的工作。
- --version　显示版本信息。

实例

显示当前时间

```
# hwclock
2010年05月27日 星期四 18时04分31秒 -0.704214 seconds
```

查看版本信息

```
# hwclock -v
hwclock from util-linux-2.12a
```

Linux apmd命令

Linux apmd命令用于进阶电源管理服务程序。

apmd负责BIOS进阶电源管理(APM)相关的记录，警告与管理工作。

语法

```
apmd [-u v V W][-p <百分比变化量>][-w <百分比值>]
```

参数：

- -p<百分比变化量>或--percentage<百分比变化量> 当电力变化的幅度超出设置的百分比变化量，即记录事件百分比变化量的预设值为5，若设置值超过100，则关闭此功能。
- -u或--utc 将BIOS时钟设为UTC，以便从悬待模式恢复时，将-u参数传送至clock或hwclock程序。
- -v或--verbose 记录所有的APM事件。
- -V或--version 显示版本信息。
- -w<百分比值>或--warn<百分比值> 当电池不在充电状态时，且电池电量低于设置的百分比值，则在syslog(2)的ALERT层记录警告信息。百分比值的预设置为10，若设置为0，则关闭此功能。
- -W或--wall 发出警告信息给所有人。

实例

记录所有的电源管理事件

```
# apmd -v
```

设置BIOS时钟

```
# apmd -utc //设置BIOS时钟为UTC
```

Linux fbset命令

Linux fbset命令用于设置景框缓冲区。

fbset指令可用于设置景框缓冲区的大小，还能调整画面之分辨率，位置，高低宽窄，色彩 深度，并可决定是否启动先卡之各项硬件特性。

语法

```
fbset [-ahinsvVx][-db <信息文件>][-fb <外围设备代号>][--test][显示模式]
```

参数：

- -a或--all 　改变所有使用该设备之虚拟终端机的显示模式。
- -db<信息文件> 　指定显示模式的信息文件，预设值文件名称为fb.modes，存放在/etc目录下
- -fb<外围设备代号> 　指定用来做为输出景框缓冲区之外围设备，预设置为"/dev/fd0"。
- -h或-help 　在线帮助。
- -i或--info 　列出所有景框缓冲区之相关信息。
- -ifb<外围设备代号> 　使用另一个景框缓冲区外围设备之设置值。
- -n或--now 　马上改变显示模式。
- -ofb<外围设备代号> 　此参数效果和指定"-fb"参数相同。
- -s或--show 　列出目前显示模式之设置。
- -v或--verbose 　显示指令执行过程。
- -V或--version 　显示版本信息。
- -x或--xfree86 　使用XFree86兼容模式。
- --test 　仅做测试，并不改变现行的显示模式。

实例

设置画面分辨率 和桌面分辨率

```
# fbset -g 800 688 1024 768//画面分辨率为800*600 桌面分辨率为1024*768
```

启动硬件文本加速

```
# fbset -accel true  // 启动硬件文本加速
```

启动广播功能

```
# fbset -bcast true //启动广播功能
```

Linux unalias命令

Linux unalias命令用于删除别名。

unalias为shell内建指令，可删除别名设置。

语法

```
unalias [-a][别名]
```

参数：

- -a 　删除全部的别名。

实例

给命令设置别名

```
[root@w3cschool.cc ~]# alias lx=ls
[root@w3cschool.cc ~]# lx
anaconda-ks.cfg Desktop install.log install.log.syslog qte
```

删除别名

```
[root@w3cschool.cc ~]# alias lx //显示别名
alias lx='ls'
[root@w3cschool.cc ~]# unalias lx //删除别名
[root@w3cschool.cc ~]# lx
-bash: lx: command not found
```

Linux SVGATextMode命令

Linux SVGATextMode命令用于加强文字模式的显示画面。

SVGATextMode可用来设置文字模式下的显示画面，包括分辨率，字体和更新频率等。

语法

```
SVGATextMode [-acdfhmnrsv][-t <配置文件>][模式]
```

参数：

- -a 如果新显示模式的屏幕大小与原先不同时，SVGATextMode会执行必要的系统设置。
- -c 维持原有的VGA时脉。
- -d 执行时会显示详细的信息，供排错时参考。
- -f 不要执行配置文件中有关字体载入的指令。
- -h 显示帮助。
- -m 允许1x1的方式来重设屏幕大小。
- -n 仅测试指定的模式。
- -r 通知或重设与屏幕大小相关的程序。
- -s 显示配置文件中所有可用的模式。
- -t<配置文件> 指定配置文件。
- -v SVGATextMode在配置新的显示模式时，预设会先检查垂直与水平的更新更新频率是否在配置文件所指定的范围内，如果不在范围内，则不设置新的显示模式。
- 模式] [模式]参数必须是配置文件中模式的名称。

Linux命令大全 - 备份压缩

ar	bunzip2	bzip2	bzip2recover
gunzip	unarj	compress	cpio
dump	uuencode	gzexe	gzip
lha	restore	tar	uudecode
unzip	zip	zipinfo	

Linux bzip2recover命令

Linux bzip2recover命令用来修复损坏的.bz2文件。

bzip2是以区块的方式来压缩文件，每个区块视为独立的单位。因此，当某一区块损坏时，便可利用bzip2recover，试着将文件中的区块隔开来，以便解压缩正常的区块。通常只适用在压缩文件很大的情况。

语法

```
bzip2recover [.bz2 压缩文件]
```

实例

修复.bz2文件

bzip2recover col.bz2

Linux bzip2命令

Linux bzip2命令是.bz2文件的压缩程序。

bzip2采用新的压缩演算法，压缩效果比传统的LZ77/LZ78压缩演算法来得好。若没有加上任何参数，bzip2压缩完文件后会产生.bz2的压缩文件，并删除原始的文件。

语法

```
bzip2 [-cdfhkLstvVz][--repetitive-best][--repetitive-fast][- 压缩等级][要压缩的文件]
```

参数：

- -c或--stdout 　将压缩与解压缩的结果送到标准输出。
- -d或--decompress 　执行解压缩。
- -f或--force 　bzip2在压缩或解压缩时，若输出文件与现有文件同名，预设不会覆盖现有文件。若要覆盖，请使用此参数。
- -h或--help 　显示帮助。
- -k或--keep 　bzip2在压缩或解压缩后，会删除原始的文件。若要保留原始文件，请使用此参数。
- -s或--small 　降低程序执行时内存的使用量。
- -t或--test 　测试.bz2压缩文件的完整性。
- -v或--verbose 　压缩或解压缩文件时，显示详细的信息。
- -z或--compress 　强制执行压缩。
- -L,--license,
- -V或--version 　显示版本信息。
- --repetitive-best 　若文件中有重复出现的资料时，可利用此参数提高压缩效果。
- --repetitive-fast 　若文件中有重复出现的资料时，可利用此参数加快执行速度。
- -压缩等级 　压缩时的区块大小。

实例

解压.bz2文件

```
[root@w3cschool.cc ~]# bzip2 -v temp.bz2 //解压文件显示详细处理信息
```

压缩文件

```
[root@w3cschool.cc ~]# bzip2 -c a.c b.c c.c
```

检查文件完整性

```
[root@w3cschool.cc ~]# bzip2 -t temp.bz2
```

Linux bunzip2命令

Linux bunzip2命令是.bz2文件的解压缩程序。

bunzip2可解压缩.bz2格式的压缩文件。bunzip2实际上是bzip2的符号连接，执行bunzip2与bzip2 -d的效果相同。

语法：bunzip2 [-fkLsvV][.bz2压缩文件]

参数：

- -f或--force 解压缩时，若输出的文件与现有文件同名时，预设不会覆盖现有的文件。若要覆盖，请使用此参数。
- -k或--keep 在解压缩后，预设会删除原来的压缩文件。若要保留压缩文件，请使用此参数。
- -s或--small 降低程序执行时，内存的使用量。
- -v或--verbose 解压缩文件时，显示详细的信息。
- -l,--license,-V或--version 显示版本信息。

实例

解压.bz2文件

```
# bunzip2 -v temp.bz2 //解压文件显示详细处理信息
```

Linux ar命令

Linux ar命令用于建立或修改备存文件，或是从备存文件中抽取文件。

ar可让您集合许多文件，成为单一的备存文件。在备存文件中，所有成员文件皆保有原来的属性与权限。

语法

```
ar[-dmpqrtx][cfosSuvV][a<成员文件>][b<成员文件>][i<成员文件>][备存文件][成员文件]
```

参数：

必要参数：

- -d 删除备存文件中的成员文件。
- -m 变更成员文件在备存文件中的次序。
- -p 显示备存文件中的成员文件内容。
- -q 将问家附加在备存文件末端。
- -r 将文件插入备存文件中。
- -t 显示备存文件中所包含的文件。
- -x 自备存文件中取出成员文件。

选项参数：

- a<成员文件> 将文件插入备存文件中指定的成员文件之后。
- b<成员文件> 将文件插入备存文件中指定的成员文件之前。
- c 建立备存文件。
- f 为避免过长的文件名不兼容于其他系统的ar指令指令，因此可利用此参数，截掉要放入备存文件中过长的成员文件名称。
- i<成员文件> 将问家插入备存文件中指定的成员文件之前。
- o 保留备存文件中文件的日期。
- s 若备存文件中包含了对象模式，可利用此参数建立备存文件的符号表。
- S 不产生符号表。
- u 只将日期较新文件插入备存文件中。
- v 程序执行时显示详细的信息。
- V 显示版本信息。

实例

打包文件

```
[root@w3cschool.cc ~]# ls   //显示当前目录文件
a.c    b.c d.c   install.log      qte
anaconda-ks.cfg c.c Desktop

[root@w3cschool.cc ~]# ar rv one.bak a.c b.c //打包 a.c b.c文件
ar: 正在创建 one.bak
a - a.c
a - b.c
[root@w3cschool.cc ~]#
```

打包多个文件

```
[root@w3cschool.cc ~]# ar rv two.bak *.c  //打包以.c结尾的文件
ar: 正在创建 two.bak
a - a.c
a - b.c
a - c.c
a - d.c
[root@w3cschool.cc ~]#
```

显示打包文件的内容

```
[root@w3cschool.cc ~]# ar t two.bak
a.c
b.c
c.c
d.c
[root@w3cschool.cc ~]#
```

删除打包文件的成员文件

```
[root@w3cschool.cc ~]# ar d two.bak a.c b.c c.c
[root@w3cschool.cc ~]# ar t two.bak
d.c
```

Linux gunzip命令

Linux gunzip命令用于解压文件。

gunzip是个使用广泛的解压缩程序，它用于解开被gzip压缩过的文件，这些压缩文件预设最后的扩展名为".gz"。事实上gunzip就是gzip的硬连接，因此不论是压缩或解压缩，都可通过gzip指令单独完成。

语法

参数：

```
gunzip [-acfhlLnNqrtvV][-s <压缩字尾字符串>][文件...] 或 gunzip [-acfhlLnNqrtvV][-s <压缩字尾
```

- -a或--ascii 　使用ASCII文字模式。
- -c或--stdout或--to-stdout 　把解压后的文件输出到标准输出设备。
- -f或-force 　强行解开压缩文件，不理会文件名称或硬连接是否存在以及该文件是否为符号连接。
- -h或--help 　在线帮助。
- -l或--list 　列出压缩文件的相关信息。
- -L或--license 　显示版本与版权信息。
- -n或--no-name 　解压缩时，若压缩文件内含有远来的文件名称及时间戳记，则将其忽略不予处理。
- -N或--name 　解压缩时，若压缩文件内含有原来的文件名称及时间戳记，则将其回存到解开的文件上。
- -q或--quiet 　不显示警告信息。
- -r或--recursive 　递归处理，将指定目录下的所有文件及子目录一并处理。
- -S<压缩字尾字符串>或--suffix<压缩字尾字符串> 　更改压缩字尾字符串。
- -t或--test 　测试压缩文件是否正确无误。
- -v或--verbose 　显示指令执行过程。
- -V或--version 显示版本信息。

实例

```
<p>解压文件
</p>
<pre>
# gunzip ab.gz
```

Linux unarj命令

Linux unarj命令用于解压缩.arj文件。

unarj为.arj压缩文件的压缩程序。

语法

```
unarj [eltx][.arj压缩文件]
```

参数：

- e 解压缩.arj文件。
- l 显示压缩文件内所包含的文件。
- t 检查压缩文件是否正确。
- x 解压缩时保留原有的路径。

实例

解压.arj文件

```
# unarj e test.arj
```

Linux compress命令

Linux compress命令是一个相当古老的 unix 档案压缩指令，压缩后的档案会加上一个 .Z 延伸档名以区别未压缩的档案，压缩后的档案可以以 uncompress 解压。若要将数个档案压成一个压缩档，必须先将档案 tar 起来再压缩。由于 gzip 可以产生更理想的压缩比例，一般人多已改用 gzip 为档案压缩工具。

语法

```
compress [-dfvcV] [-b maxbits] [file ...]
```

参数：

- c 输出结果至标准输出设备（一般指荧幕）
- f 强迫写入档案，若目的档已经存在，则会被覆盖 (force)
- v 将程序执行的讯息印在荧幕上 (verbose)
- b 设定共同字串数的上限，以位元计算，可以设定的值为 9 至 16 bits 。由于值越大，能使用的共同字串就 越多，压缩比例就越大，所以一般使用预设值 16 bits (bits)
- d 将压缩档解压缩
- V 列出版本讯息
- 范例：
- 将 source.dat 压缩成 source.dat.Z ，若 source.dat.Z 已经存在，内容则会被压缩档覆盖。
- compress -f source.dat
- 将 source.dat 压缩成 source.dat.Z ，并列印出压缩比例。
- -v 与 -f 可以一起使用
- compress -vf source.dat
- 将压缩后的资料输出后再导入 target.dat.Z 可以改变压缩档名。
- compress -c source.dat > target.dat.Z
- -b 的值越大，压缩比例就越大，范围是 9-16 ，预设值是 16 。
- compress -b 12 source.dat
- 将 source.dat.Z 解压成 source.dat ，若档案已经存在，使用者按 y 以确定覆盖档案，若使用 -df 程序则会自动覆盖档案。由于系统会自动加入 .Z 为延伸档名，所以 source.dat 会自动当作 source.dat.Z 处理。
- compress -d source.dat
- compress -d source.dat.Z

压缩文件

```
[root@w3cschool.cc ~]# compress abc.h
[root@w3cschool.cc ~]# ls

abc.h.Z
```

解压文件

```
[root@w3cschool.cc ~]# compress -d abc.h.Z
[root@w3cschool.cc ~]# ls

abc.h.
```

按指定压缩比例进行压缩

```
[root@w3cschool.cc ~]# compress -b 7 abc.h
```

强制压缩文件夹

```
[root@w3cschool.cc ~]# compress -rf /home/abc/
```

Linux cpio命令

Linux cpio命令用于备份文件。

cpio是用来建立，还原备份档的工具程序，它可以加入，解开cpio或tra备份档内的文件。

语法

```
cpio [-0aABckLovV][-C <输入/输出大小>][-F <备份档>][-H <备份格式>][-O <备份档>][--block-size=<
```

参数：

- -0或--null　接受新增列控制字符，通常配合find指令的"-print0"参数使用。
- -a或--reset-access-time　重新设置文件的存取时间。
- -A或--append　附加到已存在的备份档中，且这个备份档必须存放在磁盘上，而不能放置于磁带机里。
- -b或--swap　此参数的效果和同时指定"-sS"参数相同。
- -B　将输入/输出的区块大小改成5210 Bytes。
- -c　使用旧ASCII备份格式。
- -C<区块大小>或--io-size=<区块大小>　设置输入/输出的区块大小，单位是Byte。
- -d或--make-directories　如有需要cpio会自行建立目录。
- -E<范本文件>或--pattern-file=<范本文件>　指定范本文件，其内含有一个或多个范本样式，让cpio解开符合范本条件的文件，格式为每列一个范本样式。
- -f或--nonmatching　让cpio解开所有不符合范本条件的文件。
- -F<备份档>或--file=<备份档>　指定备份档的名称，用来取代标准输入或输出，也能借此通过网络使用另一台主机的保存设备存取备份档。
- -H<备份格式>　指定备份时欲使用的文件格式。
- -i或--extract　执行copy-in模式，还原备份档。
- -I<备份档>　指定备份档的名称，用来取代标准输入，也能借此通过网络使用另一台主机的保存设备读取备份档。
- -k　此参数将忽略不予处理，仅负责解决cpio不同版本间的兼容性问题。
- -l或--link　以硬连接的方式取代复制文件，可在copy-pass模式下运用。
- -L或--dereference　不建立符号连接，直接复制该连接所指向的原始文件。
- -m或preserve-modification-time　不去更换文件的更改时间。
- -M<回传信息>或--message=<回传信息>　设置更换保存媒体的信息。
- -n或--numeric-uid-gid　使用"-tv"参数列出备份档的内容时，若再加上参数"-n"，则会以用户识别码和群组识别码替代拥有者和群组名称列出文件清单。
- -o或--create　执行copy-out模式，建立备份档。

- -O<备份档> 指定备份档的名称，用来取代标准输出，也能借此通过网络 使用另一台主机的保存设备存放备份档。
- -p或--pass-through 执行copy-pass模式，略过备份步骤，直接将文件复制到目的目录。
- -r或--rename 当有文件名称需要更动时，采用互动模式。
- -R<拥有者><:/.><所属群组>或
- ----owner<拥有者><:/.><所属群组> 在copy-in模式还原备份档，或copy-pass模式复制文件时，可指定这些备份，复制的文件的拥有者与所属群组。
- -s或--swap-bytes 交换每对字节的内容。
- -S或--swap-halfwords 交换每半个字节的内容。
- -t或--list 将输入的内容呈现出来。
- -u或--unconditional 置换所有文件，不论日期时间的新旧与否，皆不予询问而直接覆盖。
- -v或--verbose 详细显示指令的执行过程。
- -V或--dot 执行指令时，在每个文件的执行程序前面加上"."号
- --block-size=<区块大小> 设置输入/输出的区块大小，假如设置数值为5，则区块大小为2500，若设置成10，则区块大小为5120，依次类推。
- --force-local 强制将备份档存放在本地主机。
- --help 在线帮助。
- --no-absolute-filenames 使用相对路径建立文件名称。
- --no-preserve-owner 不保留文件的拥有者，谁解开了备份档，那些文件就归谁所有。
- -only-verify-crc 当备份档采用CRC备份格式时，可使用这项参数检查备份档内的每个文件是否正确无误。
- --quiet 不显示复制了多少区块。
- --sparse 倘若一个文件内含大量的连续0字节，则将此文件存成稀疏文件。
- --version 显示版本信息。

实例

制作备份文件

```
[root@w3cschool.cc var]# ll //显示当前目录下的文件
总用量 164
drwxr-xr-x  2 root  root  4096 2008-03-30 account
drwxr-xr-x  9 root  root  4096 2008-03-30 cache
drwxr-xr-x  3 netdump netdump 4096 2008-03-30 crash
drwxr-xr-x  3 root  root  4096 2008-03-30 db
drwxr-xr-x  3 root  root  4096 2008-03-30 empty
drwxr-xr-x  3 root  root  4096 2008-03-30 ftp
drwxrwx--T  2 root  gdm   4096 4月 9 20:17 gdm
drwxr-xr-x 25 root  root  4096 2008-03-30 lib
drwxr-xr-x  2 root  root  4096 2004-08-13 local
drwxrwxr-x  6 root  lock  4096 5月 8 15:25 lock
drwxr-xr-x 14 root  root  4096 5月 8 15:14 log
lrwxrwxrwx  1 root  root   10 2008-03-30 mail -> spool/mail
drwxr-xr-x  2 root  root  4096 2004-08-13 nis
drwxr-xr-x  2 root  root  4096 2004-08-13 opt
drwxr-xr-x  2 root  root  4096 2004-08-13 preserve
drwxr-xr-x 16 root  root  4096 5月 8 15:14 run
drwxr-xr-x 16 root  root  4096 2008-03-30 spool
drwxrwxrwt  3 root  root  4096 1月 13 18:53 tmp
drwx------  2 root  root  4096 2004-07-08 tux
drwxr-xr-x  8 root  root  4096 1月 19 19:39 www
drwxr-xr-x  3 root  root  4096 2008-03-30 yp
[root@w3cschool.cc var]# ls | cpio -o >123.cpio //制作备份文件
25 blocks
[root@w3cschool.cc var]# ll //显示当前目录下的文件
总用量 172
-rw-r--r--  1 root  root  1024 5月 24 13:06 123.cpio
drwxr-xr-x  2 root  root  4096 2008-03-30 account
drwxr-xr-x  9 root  root  4096 2008-03-30 cache
drwxr-xr-x  3 netdump netdump 4096 2008-03-30 crash
drwxr-xr-x  3 root  root  4096 2008-03-30 db
drwxr-xr-x  3 root  root  4096 2008-03-30 empty
drwxr-xr-x  3 root  root  4096 2008-03-30 ftp
drwxrwx--T  2 root  gdm   4096 4月 9 20:17 gdm
drwxr-xr-x 25 root  root  4096 2008-03-30 lib
drwxr-xr-x  2 root  root  4096 2004-08-13 local
drwxrwxr-x  6 root  lock  4096 5月 8 15:25 lock
drwxr-xr-x 14 root  root  4096 5月 8 15:14 log
lrwxrwxrwx  1 root  root   10 2008-03-30 mail -> spool/mail
drwxr-xr-x  2 root  root  4096 2004-08-13 nis
drwxr-xr-x  2 root  root  4096 2004-08-13 opt
drwxr-xr-x  2 root  root  4096 2004-08-13 preserve
drwxr-xr-x 16 root  root  4096 5月 8 15:14 run
drwxr-xr-x 16 root  root  4096 2008-03-30 spool
drwxrwxrwt  3 root  root  4096 1月 13 18:53 tmp
drwx------  2 root  root  4096 2004-07-08 tux
drwxr-xr-x  8 root  root  4096 1月 19 19:39 www
drwxr-xr-x  3 root  root  4096 2008-03-30 yp
[root@w3cschool.cc var]#
```

解压备份文件

```
[root@w3cschool.cc var]# ls | cpio -i -l 123.cpio
```

解压缩备份文件，并列出详细信息

```
[root@w3cschool.cc var]# cpio -t -I 123.cpio
123.cpio
a.c
b.c
c.c
……省略部分结果
```

强制解压缩

```
[root@w3cschool.cc var]# cpio -i -u -I 123.cpio
```

解压缩时进行反向匹配，指定不解压的文件

```
[root@w3cschool.cc var]# cpio -i -I 123.cpio -f *.c

//不解压.c结尾的文件
```

向指定的.cpio文件添加文件

```
[root@w3cschool.cc var]# ls
123.cpio crash ftp local mail preserve tmp yp
account  db    gdm lock  nis  run       tux
cache      empty lib log  opt  spool   www
[root@w3cschool.cc var]# cpio -o -O 123.cpio -A
db  //用户输入 按下Ctrl+D结束输入
1 block
[root@w3cschool.cc var]#
```

从标准输入备份文件

```
[root@w3cschool.cc test]# ls
a. a.c b.c c.c d.c    f.c
[root@w3cschool.cc test]# cpio -o >123.cpio
a.c  //用户输入
b.c
c.c  //按下Ctrl+D完成输入
3 block
[root@w3cschool.cc test]#
```

复制文件

```
[root@w3cschool.cc test]# cpio -p /root
a.c  //用户输入
b.c
c.c  //按下Ctrl+D完成输入
3 block
```

Linux dump命令

Linux dump命令用于备份文件系统。

dump为备份工具程序，可将目录或整个文件系统备份至指定的设备，或备份成一个大文件。

语法

```
dump [-cnu][-0123456789][-b <区块大小>][-B <区块数目>][-d <密度>][-f <设备名称>][-h <层级>][-s
```

参数：

- -0123456789　备份的层级。
- -b<区块大小>　指定区块的大小，单位为KB。
- -B<区块数目>　指定备份卷册的区块数目。
- -c　修改备份磁带预设的密度与容量。
- -d<密度>　设置磁带的密度。单位为BPI。
- -f<设备名称>　指定备份设备。
- -h<层级>　当备份层级等于或大雨指定的层级时，将不备份用户标示为"nodump"的文件。
- -n　当备份工作需要管理员介入时，向所有"operator"群组中的使用者发出通知。
- -s<磁带长度>　备份磁带的长度，单位为英尺。
- -T<日期>　指定开始备份的时间与日期。
- -u　备份完毕后，在/etc/dumpdates中记录备份的文件系统，层级，日期与时间等。
- -w　与-W类似，但仅显示需要备份的文件。
- -W　显示需要备份的文件及其最后一次备份的层级，时间与日期。

实例

备份文件到磁带

```
# dump -0 -u /dev/tape /home/
```

其中"-0"参数指定的是备份等级"-u"要求备份完毕之后将相应的信息存储到文件 /etc/dumpdates 留作记录

Linux uuencode命令

Linux uuencode命令用于将uuencode编码后的档案还原。

早期在许多 unix 系统的传送协定只能传送七位元字元，并不支援二进位档案，像中文文字档就有用到八位元，所以无法完整地送到另一架机器上。 uuencode 指令，可以将二进位档转换成七位元的档案，传送到另一架机器上再以 uudecode 还原。最常见的是用在以电子邮件传送二进位档。uuencode 编码后的资料都以 begin 开始，以 end 作为结束。

语法

```
compress[必要参数][选择参数][目录或者文件]
```

参数说明：

必要参数：

- 无

选择参数：

- h 显示帮助信息
- v 显示版本信息

实例

还原档案

```
# uuencode test.uud
```

Linux restore命令

Linux restore命令用来还原由dump操作所备份下来的文件或整个文件系统(一个分区)。

restore 指令所进行的操作和dump指令相反，dump操作可用来备份文件，而restore操作则是写回这些已备份的文件。

语法

```
restore [-cCvy][-b <区块大小>][-D <文件系统>][-f <备份文件>][-s <文件编号>] 或 restore [-chimvy
```

参数：

- -b<区块大小> 设置区块大小，单位是Byte。
- -c 不检查dump操作的备份格式，仅准许读取使用旧格式的备份文件。
- -C 使用对比模式，将备份的文件与现行的文件相互对比。
- -D<文件系统> 允许用户指定文件系统的名称。
- -f<备份文件> 从指定的文件中读取备份数据，进行还原操作。
- -h 仅解出目录而不包括与该目录相关的所有文件。
- -i 使用互动模式，在进行还原操作时，restore指令将依序询问用户。
- -m 解开符合指定的inode编号的文件或目录而非采用文件名称指定。
- -r 进行还原操作。
- -R 全面还原文件系统时，检查应从何处开始进行。
- -s<文件编号> 当备份数据超过一卷磁带时，您可以指定备份文件的编号。
- -t 指定文件名称，若该文件已存在备份文件中，则列出它们的名称。
- -v 显示指令执行过程。
- -x 设置文件名称，且从指定的存储媒体里读入它们，若该文件已存在在备份文件中，则将其还原到文件系统内。
- -y 不询问任何问题，一律以同意回答并继续执行指令。

Linux lha命令

Linux lha命令用于压缩或解压缩文件。

lha是从lharc演变而来的压缩程序，文件经它压缩后，会另外产生具有".lzh"扩展名的压缩文件。

语法

```
lha [-acdfglmnpqtuvx][-a <0/1/2>/u</0/1/2>][-<a/c/u>d][-<e/x>i][-<a/u>o][-<e/x>w=<目的目录
```

参数：

- -a或a 压缩文件，并加入到压缩文件内。
- -a<0/1/2>/u</0/1/2> 压缩文件时，采用不同的文件头。
- -c或c 压缩文件，重新建构新的压缩文件后，再将其加入。
- -d或d 从压缩文件内删除指定的文件。
- -<a/c/u>d或<a/c/u>d 压缩文件，然后将其加入，重新建构，更新压缩文件或，删除原始文件，也就是把文件移到压缩文件中。
- -e或e 解开压缩文件。
- -f或f 强制执行lha命令，在解压时会直接覆盖已有的文件而不加以询问。
- -g或g 使用通用的压缩格式，便于解决兼容性的问题。
- -<e/x>i或<e/x>i 解开压缩文件时，忽略保存在压缩文件内的文件路径，直接将其解压后存放在现行目录下或是指定的目录中。
- -l或l 列出压缩文件的相关信息。
- -m或m 此参数的效果和同时指定"-ad"参数相同。
- -n或n 不执行指令，仅列出实际执行会进行的动作。
- -<a/u>o或<a/u>o 采用lharc兼容格式，将压缩后的文件加入，更新压缩文件。
- -p或p 从压缩文件内输出到标准输出设备。
- -q或q 不显示指令执行过程。
- -t或t 检查备份文件内的每个文件是否正确无误。
- -u或u 更换较新的文件到压缩文件内。
- -u</0/1/2>或u</0/1/2> 在文件压缩时采用不同的文件头，然后更新到压缩文件内。
- -v或v 详细列出压缩文件的相关信息。
- -<e/x>w=<目的目录>或<e/x>w=<目的目录> 指定解压缩的目录。
- -x或x 解开压缩文件。
- -<a/u>z或<a/u>z 不压缩文件，直接把它加入，更新压缩文件。

实例

缩文件

```
# lha -a abc.lhz a.b //压缩a.b文件，压缩后生成 abc.lhz文件
```

压缩目录

```
# lha -a abc2 /home/hnlinux
```

解压文件到当前目录

```
# lha -xiw=agis abc  //解压文件abc
```

Linux gzip命令

Linux gzip命令用于压缩文件。

gzip是个使用广泛的压缩程序，文件经它压缩过后，其名称后面会多出".gz"的扩展名。

语法

```
gzip [-acdfhlLnNqrtvV][-S &lt;压缩字尾字符串&gt;][-&lt;压缩效率&gt;][--best/fast][文件...] 或
```

参数：

- -a或--ascii 使用ASCII文字模式。
- -c或--stdout或--to-stdout 把压缩后的文件输出到标准输出设备，不去更动原始文件。
- -d或--decompress或----uncompress 解开压缩文件。
- -f或--force 强行压缩文件。不理会文件名称或硬连接是否存在以及该文件是否为符号连接。
- -h或--help 在线帮助。
- -l或--list 列出压缩文件的相关信息。
- -L或--license 显示版本与版权信息。
- -n或--no-name 压缩文件时，不保存原来的文件名称及时间戳记。
- -N或--name 压缩文件时，保存原来的文件名称及时间戳记。
- -q或--quiet 不显示警告信息。
- -r或--recursive 递归处理，将指定目录下的所有文件及子目录一并处理。
- -S<压缩字尾字符串>或----suffix<压缩字尾字符串> 更改压缩字尾字符串。
- -t或--test 测试压缩文件是否正确无误。
- -v或--verbose 显示指令执行过程。
- -V或--version 显示版本信息。
- -<压缩效率> 压缩效率是一个介于1－9的数值，预设值为"6"，指定愈大的数值，压缩效率就会愈高。
- --best 此参数的效果和指定"-9"参数相同。
- --fast 此参数的效果和指定"-1"参数相同。

实例

压缩文件

```
[root@w3cschool.cc a]# ls //显示当前目录文件
a.c b.h d.cpp
[root@w3cschool.cc a]# gzip * //压缩目录下的所有文件
[root@w3cschool.cc a]# ls //显示当前目录文件
a.c.gz    b.h.gz    d.cpp.gz
[root@w3cschool.cc a]#
```

接范例1， 列出详细的信息

```
[root@w3cschool.cc a]# gzip -dv * //解压文件，并列出详细信息
a.c.gz:     0.0% -- replaced with a.c
b.h.gz:     0.0% -- replaced with b.h
d.cpp.gz:     0.0% -- replaced with d.cpp
[root@w3cschool.cc a]#
```

接范例1，显示压缩文件的信息

```
[root@w3cschool.cc a]# gzip -l *
     compressed    uncompressed ratio uncompressed_name
         24          0  0.0% a.c
         24          0  0.0% b.h
         26          0  0.0% d.cpp
```

Linux gzexe命令

Linux gzexe命令用于压缩执行文件。

gzexe是用来压缩执行文件的程序。当您去执行被压缩过的执行文件时，该文件会自动解压然后继续执行，和使用一般的执行文件相同。

语法

```
gzexe [-d][执行文件...]
```

参数：

- -d 解开压缩文件。

实例

压缩可执行文件

```
# gzexe abc
```

Linux zipinfo命令

Linux zipinfo命令用于列出压缩文件信息。

执行zipinfo指令可得知zip压缩文件的详细信息。

语法

```
zipinfo [-12hlmMstTvz][压缩文件][文件...][-x <范本样式>]
```

参数：

- -1 只列出文件名称。
- -2 此参数的效果和指定"-1"参数类似，但可搭配"-h","-t"和"-z"参数使用。
- -h 只列出压缩文件的文件名称。
- -l 此参数的效果和指定"-m"参数类似，但会列出原始文件的大小而非每个文件的压缩率。
- -m 此参数的效果和指定"-s"参数类似，但多会列出每个文件的压缩率。
- -M 若信息内容超过一个画面，则采用类似more指令的方式列出信息。
- -s 用类似执行"ls -l"指令的效果列出压缩文件内容。
- -t 只列出压缩文件内所包含的文件数目，压缩前后的文件大小及压缩率。
- -T 将压缩文件内每个文件的日期时间用年，月，日，时，分，秒的顺序列出。
- -v 详细显示压缩文件内每一个文件的信息。
- -x<范本样式> 不列出符合条件的文件的信息。
- -z 如果压缩文件内含有注释，就将注释显示出来。

实例

显示压缩文件信息

```
[root@w3cschool.cc a]# zipinfo cp.zip
Archive: cp.zip  486 bytes  4 files
-rw-r--r-- 2.3 unx    0 bx stor 24-May-10 18:54 a.c
-rw-r--r-- 2.3 unx    0 bx stor 24-May-10 18:54 b.c
-rw-r--r-- 2.3 unx    0 bx stor 24-May-10 18:54 c.c
-rw-r--r-- 2.3 unx    0 bx stor 24-May-10 18:54 e.c
4 files, 0 bytes uncompressed, 0 bytes compressed: 0.0%
[root@w3cschool.cc a]#
```

显示压缩文件中每个文件的信息

```
[root@w3cschool.cc a]# zipinfo -v cp.zip
Archive: cp.zip  486 bytes  4 files

End-of-central-directory record:
```

```
-------------------------------

  Actual offset of end-of-central-dir record:     464 (000001D0h)
  Expected offset of end-of-central-dir record:    464 (000001D0h)
  (based on the length of the central directory and its expected offset)

  This zipfile constitutes the sole disk of a single-part archive; its
  central directory contains 4 entries. The central directory is 248
  (000000F8h) bytes long, and its (expected) offset in bytes from the
  beginning of the zipfile is 216 (000000D8h).

  There is no zipfile comment.

Central directory entry #1:
---------------------------

  a.c

  offset of local header from start of archive:   0 (00000000h) bytes
  file system or operating system of origin:    Unix
  version of encoding software:           2.3
  minimum file system compatibility required:    MS-DOS, OS/2 or NT FAT
  minimum software version required to extract:   1.0
  compression method:                none (stored)
  file security status:               not encrypted
  extended local header:              no
  file last modified on (DOS date/time):      2010 May 24 18:54:26
  file last modified on (UT extra field modtime):  2010 May 24 18:54:26 local
  file last modified on (UT extra field modtime):  2010 May 24 10:54:26 UTC
  32-bit CRC value (hex):              00000000
  compressed size:                 0 bytes
  uncompressed size:                0 bytes
  length of filename:                3 characters
  length of extra field:              13 bytes
  length of file comment:              0 characters
  disk number on which file begins:         disk 1
  apparent file type:                binary
  Unix file attributes (100644 octal):       -rw-r--r--
  MS-DOS file attributes (00 hex):         none

  The central-directory extra field contains:
  - A subfield with ID 0x5455 (universal time) and 5 data bytes.
   The local extra field has UTC/GMT modification/access times.
  - A subfield with ID 0x7855 (Unix UID/GID) and 0 data bytes.

  There is no file comment.

Central directory entry #2:
---------------------------

  b.c

  offset of local header from start of archive:   54 (00000036h) bytes
  file system or operating system of origin:    Unix
  version of encoding software:           2.3
  minimum file system compatibility required:    MS-DOS, OS/2 or NT FAT
  minimum software version required to extract:   1.0
  compression method:                none (stored)
  file security status:               not encrypted
  extended local header:              no
  file last modified on (DOS date/time):      2010 May 24 18:54:26
  file last modified on (UT extra field modtime):  2010 May 24 18:54:26 local
  file last modified on (UT extra field modtime):  2010 May 24 10:54:26 UTC
  32-bit CRC value (hex):              00000000
  compressed size:                 0 bytes
  uncompressed size:                0 bytes
  length of filename:                3 characters
  length of extra field:              13 bytes
  length of file comment:              0 characters
  disk number on which file begins:         disk 1
  apparent file type:                binary
  Unix file attributes (100644 octal):       -rw-r--r--
```

```
  MS-DOS file attributes (00 hex):         none

  The central-directory extra field contains:
  - A subfield with ID 0x5455 (universal time) and 5 data bytes.
   The local extra field has UTC/GMT modification/access times.
  - A subfield with ID 0x7855 (Unix UID/GID) and 0 data bytes.

  There is no file comment.

Central directory entry #3:
---------------------------

  c.c

  offset of local header from start of archive:   108 (0000006Ch) bytes
  file system or operating system of origin:    Unix
  version of encoding software:           2.3
  minimum file system compatibility required:    MS-DOS, OS/2 or NT FAT
  minimum software version required to extract:   1.0
  compression method:                none (stored)
  file security status:               not encrypted
  extended local header:              no
  file last modified on (DOS date/time):      2010 May 24 18:54:26
  file last modified on (UT extra field modtime):  2010 May 24 18:54:26 local
  file last modified on (UT extra field modtime):  2010 May 24 10:54:26 UTC
  32-bit CRC value (hex):              00000000
  compressed size:                 0 bytes
  uncompressed size:                0 bytes
  length of filename:                3 characters
  length of extra field:              13 bytes
  length of file comment:              0 characters
  disk number on which file begins:         disk 1
  apparent file type:                binary
  Unix file attributes (100644 octal):       -rw-r--r--
  MS-DOS file attributes (00 hex):         none

  The central-directory extra field contains:
  - A subfield with ID 0x5455 (universal time) and 5 data bytes.
   The local extra field has UTC/GMT modification/access times.
  - A subfield with ID 0x7855 (Unix UID/GID) and 0 data bytes.

  There is no file comment.

Central directory entry #4:
---------------------------

  e.c

  offset of local header from start of archive:   162 (000000A2h) bytes
  file system or operating system of origin:    Unix
  version of encoding software:           2.3
  minimum file system compatibility required:    MS-DOS, OS/2 or NT FAT
  minimum software version required to extract:   1.0
  compression method:                none (stored)
  file security status:               not encrypted
  extended local header:              no
  file last modified on (DOS date/time):      2010 May 24 18:54:26
  file last modified on (UT extra field modtime):  2010 May 24 18:54:26 local
  file last modified on (UT extra field modtime):  2010 May 24 10:54:26 UTC
  32-bit CRC value (hex):              00000000
  compressed size:                 0 bytes
  uncompressed size:                0 bytes
  length of filename:                3 characters
  length of extra field:              13 bytes
  length of file comment:              0 characters
  disk number on which file begins:         disk 1
  apparent file type:                binary
  Unix file attributes (100644 octal):       -rw-r--r--
  MS-DOS file attributes (00 hex):         none

  The central-directory extra field contains:
  - A subfield with ID 0x5455 (universal time) and 5 data bytes.
```

```
 The local extra field has UTC/GMT modification/access times.
- A subfield with ID 0x7855 (Unix UID/GID) and 0 data bytes.

There is no file comment.
```

Linux zip命令

Linux zip命令用于压缩文件。

zip是个使用广泛的压缩程序，文件经它压缩后会另外产生具有".zip"扩展名的压缩文件。

语法

```
zip [-AcdDfFghjJKlLmoqrSTuvVwXyz$][-b <工作目录>][-ll][-n <字尾字符串>][-t <日期时间>][-<压缩交
```

参数：

- -A 调整可执行的自动解压缩文件。
- -b<工作目录> 指定暂时存放文件的目录。
- -c 替每个被压缩的文件加上注释。
- -d 从压缩文件内删除指定的文件。
- -D 压缩文件内不建立目录名称。
- -f 此参数的效果和指定"-u"参数类似，但不仅更新既有文件，如果某些文件原本不存在于压缩文件内，使用本参数会一并将其加入压缩文件中。
- -F 尝试修复已损坏的压缩文件。
- -g 将文件压缩后附加在既有的压缩文件之后，而非另行建立新的压缩文件。
- -h 在线帮助。
- -i<范本样式> 只压缩符合条件的文件。
- -j 只保存文件名称及其内容，而不存放任何目录名称。
- -J 删除压缩文件前面不必要的数据。
- -k 使用MS-DOS兼容格式的文件名称。
- -l 压缩文件时，把LF字符置换成LF+CR字符。
- -ll 压缩文件时，把LF+CR字符置换成LF字符。
- -L 显示版权信息。
- -m 将文件压缩并加入压缩文件后，删除原始文件，即把文件移到压缩文件中。
- -n<字尾字符串> 不压缩具有特定字尾字符串的文件。
- -o 以压缩文件内拥有最新更改时间的文件为准，将压缩文件的更改时间设成和该文件相同。
- -q 不显示指令执行过程。
- -r 递归处理，将指定目录下的所有文件和子目录一并处理。
- -S 包含系统和隐藏文件。
- -t<日期时间> 把压缩文件的日期设成指定的日期。
- -T 检查备份文件内的每个文件是否正确无误。

- -u 更换较新的文件到压缩文件内。
- -v 显示指令执行过程或显示版本信息。
- -V 保存VMS操作系统的文件属性。
- -w 在文件名称里假如版本编号，本参数仅在VMS操作系统下有效。
- -x<范本样式> 压缩时排除符合条件的文件。
- -X 不保存额外的文件属性。
- -y 直接保存符号连接，而非该连接所指向的文件，本参数仅在UNIX之类的系统下有效。
- -z 替压缩文件加上注释。
- -$ 保存第一个被压缩文件所在磁盘的卷册名称。
- -<压缩效率> 压缩效率是一个介于1-9的数值。

实例

压缩文件

```
# zip -v cp.zip a.c b.c c.c e.c
 adding: a.c    (in=0) (out=0) (stored 0%)
 adding: b.c    (in=0) (out=0) (stored 0%)
 adding: c.c    (in=0) (out=0) (stored 0%)
 adding: e.c    (in=0) (out=0) (stored 0%)
total bytes=0, compressed=0 -> 0% savings
```

压缩文件

```
# [root@ubuntu a]# zip -v cp2.zip *

#
```

压缩目录

```
# zip -r cp3.zip /root/
```

从压缩文件中删除文件

```
# zip -dv cp.zip a.c
```

Linux unzip命令

Linux unzip命令用于解压缩zip文件

unzip为.zip压缩文件的解压缩程序。

语法

```
unzip [-cflptuvz][-agCjLMnoqsVX][-P <密码>][.zip文件][文件][-d <目录>][-x <文件>] 或 unzip [-
```

参数：

- -c 将解压缩的结果显示到屏幕上，并对字符做适当的转换。
- -f 更新现有的文件。
- -l 显示压缩文件内所包含的文件。
- -p 与-c参数类似，会将解压缩的结果显示到屏幕上，但不会执行任何的转换。
- -t 检查压缩文件是否正确。
- -u 与-f参数类似，但是除了更新现有的文件外，也会将压缩文件中的其他文件解压缩到目录中。
- -v 执行是时显示详细的信息。
- -z 仅显示压缩文件的备注文字。
- -a 对文本文件进行必要的字符转换。
- -b 不要对文本文件进行字符转换。
- -C 压缩文件中的文件名称区分大小写。
- -j 不处理压缩文件中原有的目录路径。
- -L 将压缩文件中的全部文件名改为小写。
- -M 将输出结果送到more程序处理。
- -n 解压缩时不要覆盖原有的文件。
- -o 不必先询问用户，unzip执行后覆盖原有文件。
- -P<密码> 使用zip的密码选项。
- -q 执行时不显示任何信息。
- -s 将文件名中的空白字符转换为底线字符。
- -V 保留VMS的文件版本信息。
- -X 解压缩时同时回存文件原来的UID/GID。
- [.zip文件] 指定.zip压缩文件。
- [文件] 指定要处理.zip压缩文件中的哪些文件。
- -d<目录> 指定文件解压缩后所要存储的目录。
- -x<文件> 指定不要处理.zip压缩文件中的哪些文件。

- -Z unzip -Z等于执行zipinfo指令。

实例

显示压缩文件信息

```
# unzip -l abc.zip
Archive: abc.zip
 Length   Date  Time  Name
--------  ----  ----  ----
  94618 05-21-10 20:44  a11.jpg
  202001 05-21-10 20:44  a22.jpg
    16 05-22-10 15:01  11.txt
  46468 05-23-10 10:30  w456.JPG
  140085 03-14-10 21:49  my.asp
--------          -------
  483188          5 files
```

解压文件

```
# unzip -v abc.zip
Archive: abc.zip
Length  Method  Size Ratio  Date  Time  CRC-32  Name
-------- ------ ------- -----  ----  ----  ------  ----
  94618 Defl:N  93353  1% 05-21-10 20:44 9e661437 a11.jpg
 202001 Defl:N  201833  0% 05-21-10 20:44 1da462eb a22.jpg
   16 Stored    16  0% 05-22-10 15:01 ae8a9910 ? +-|￥+-? (11).txt
  46468 Defl:N  39997 14% 05-23-10 10:30 962861f2 w456.JPG
 140085 Defl:N  36765 74% 03-14-10 21:49 836fcc3f my.asp
--------     ------- ---             -------
 483188      371964 23%              5 files
```

Linux uudecode命令

Linuxuudecode 将 uuencode 编码后的档案还原， uudecode 只会将 begin 与 end 标记之间的编码资料还原，程序会跳过标记以外的资料。

语法

```
uuencode [-hv] [file1 ...]</p>
```

参数：

- h 列出指令使用格式 (help)
- v 列出版本讯息

实例

将 file.uud 还原，而还原后的档名储存在 file.uud 档中。

```
uuencode file.uud
```

可以一起还原好几个档案。

```
uuencode file1.uud file2.uud
```

Linux tar命令

Linux tar命令用于备份文件。

tar是用来建立，还原备份文件的工具程序，它可以加入，解开备份文件内的文件。

语法

```
tar [-ABcdgGhiklmMoOpPrRsStuUvwWxzZ][-b <区块数目>][-C <目的目录>][-f <备份文件>][-F <Script文
```

参数：

- -A或--catenate 新增温暖件到已存在的备份文件。
- -b<区块数目>或--blocking-factor=<区块数目> 设置每笔记录的区块数目，每个区块大小为12Bytes。
- -B或--read-full-records 读取数据时重设区块大小。
- -c或--create 建立新的备份文件。
- -C<目的目录>或--directory=<目的目录> 切换到指定的目录。
- -d或--diff或--compare 对比备份文件内和文件系统上的文件的差异。
- -f<备份文件>或--file=<备份文件> 指定备份文件。
- -F<Script文件>或--info-script=<Script文件> 每次更换磁带时，就执行指定的Script文件。
- -g或--listed-incremental 处理GNU格式的大量备份。
- -G或--incremental 处理旧的GNU格式的大量备份。
- -h或--dereference 不建立符号连接，直接复制该连接所指向的原始文件。
- -i或--ignore-zeros 忽略备份文件中的0 Byte区块，也就是EOF。
- -k或--keep-old-files 解开备份文件时，不覆盖已有的文件。
- -K<文件>或--starting-file=<文件> 从指定的文件开始还原。
- -l或--one-file-system 复制的文件或目录存放的文件系统，必须与tar指令执行时所处的文件系统相同，否则不予复制。
- -L<媒体容量>或-tape-length=<媒体容量> 设置存放每体的容量，单位以1024 Bytes计算。
- -m或--modification-time 还原文件时，不变更文件的更改时间。
- -M或--multi-volume 在建立，还原备份文件或列出其中的内容时，采用多卷册模式。
- -N<日期格式>或--newer=<日期时间> 只将较指定日期更新的文件保存到备份文件里。
- -o或--old-archive或--portability 将资料写入备份文件时使用V7格式。
- -O或--stdout 把从备份文件里还原的文件输出到标准输出设备。
- -p或--same-permissions 用原来的文件权限还原文件。
- -P或--absolute-names 文件名使用绝对名称，不移除文件名称前的"/"号。

- -r或--append 新增文件到已存在的备份文件的结尾部分。
- -R或--block-number 列出每个信息在备份文件中的区块编号。
- -s或--same-order 还原文件的顺序和备份文件内的存放顺序相同。
- -S或--sparse 倘若一个文件内含大量的连续0字节，则将此文件存成稀疏文件。
- -t或--list 列出备份文件的内容。
- -T<范本文件>或--files-from=<范本文件> 指定范本文件，其内含有一个或多个范本样式，让tar解开或建立符合设置条件的文件。
- -u或--update 仅置换较备份文件内的文件更新的文件。
- -U或--unlink-first 解开压缩文件还原文件之前，先解除文件的连接。
- -v或--verbose 显示指令执行过程。
- -V<卷册名称>或--label=<卷册名称> 建立使用指定的卷册名称的备份文件。
- -w或--interactive 遭遇问题时先询问用户。
- -W或--verify 写入备份文件后，确认文件正确无误。
- -x或--extract或--get 从备份文件中还原文件。
- -X<范本文件>或--exclude-from=<范本文件> 指定范本文件，其内含有一个或多个范本样式，让ar排除符合设置条件的文件。
- -z或--gzip或--ungzip 通过gzip指令处理备份文件。
- -Z或--compress或--uncompress 通过compress指令处理备份文件。
- -<设备编号><存储密度> 设置备份用的外围设备编号及存放数据的密度。
- --after-date=<日期时间> 此参数的效果和指定"-N"参数相同。
- --atime-preserve 不变更文件的存取时间。
- --backup=<备份方式>或--backup 移除文件前先进行备份。
- --checkpoint 读取备份文件时列出目录名称。
- --concatenate 此参数的效果和指定"-A"参数相同。
- --confirmation 此参数的效果和指定"-w"参数相同。
- --delete 从备份文件中删除指定的文件。
- --exclude=<范本样式> 排除符合范本样式的问家。
- --group=<群组名称> 把加入设备文件中的文件的所属群组设成指定的群组。
- --help 在线帮助。
- --ignore-failed-read 忽略数据读取错误，不中断程序的执行。
- --new-volume-script=<Script文件> 此参数的效果和指定"-F"参数相同。
- --newer-mtime 只保存更改过的文件。
- --no-recursion 不做递归处理，也就是指定目录下的所有文件及子目录不予处理。
- --null 从null设备读取文件名称。
- --numeric-owner 以用户识别码及群组识别码取代用户名称和群组名称。
- --owner=<用户名称> 把加入备份文件中的文件的拥有者设成指定的用户。
- --posix 将数据写入备份文件时使用POSIX格式。
- --preserve 此参数的效果和指定"-ps"参数相同。
- --preserve-order 此参数的效果和指定"-A"参数相同。
- --preserve-permissions 此参数的效果和指定"-p"参数相同。

- --record-size=<区块数目> 此参数的效果和指定"-b"参数相同。
- --recursive-unlink 解开压缩文件还原目录之前，先解除整个目录下所有文件的连接。
- --remove-files 文件加入备份文件后，就将其删除。
- --rsh-command=<执行指令> 设置要在远端主机上执行的指令，以取代rsh指令。
- --same-owner 尝试以相同的文件拥有者还原问家你。
- --suffix=<备份字尾字符串> 移除文件前先行备份。
- --totals 备份文件建立后，列出文件大小。
- --use-compress-program=<执行指令> 通过指定的指令处理备份文件。
- --version 显示版本信息。
- --volno-file=<编号文件> 使用指定文件内的编号取代预设的卷册编号。

实例

压缩文件 非打包

```
# touch a.c
# tar -czvf test.tar.gz a.c   //压缩 a.c文件为test.tar.gz
a.c
```

列出压缩文件内容

```
# tar -tzvf test.tar.gz
-rw-r--r-- root/root     0 2010-05-24 16:51:59 a.c
```

解压文件

tar -xzvf test.tar.gz a.c

Linux命令大全 - 设备管理

setleds	loadkeys	rdev	dumpkeys
MAKEDEV			

Linux setleds命令

Linux setleds命令用来设定键盘上方三个 LED 的状态。在 Linux 中，每一个虚拟主控台都有独立的设定。

语法

```
setleds [-v] [-L] [-D] [-F] [{+|-}num] [{+|-}caps] [{+|-}scroll]
```

参数：

- -F：预设的选项，设定虚拟主控台的状态。
- -D：除了改变虚拟主控台的状态外，还改变预设的状态。
- -L：不改变虚拟主控台的状态，但直接改变 LED 显示的状态。这会使得 LDE 显示和目前虚拟主控台的状态不符合。我们可以在稍后用 -L 且不含其它选项的 setleds 命令回复正常状态。
- -num +num：将数字键打开或关闭。
- -caps +caps：把大小写键打开或关闭。
- -scroll +scroll：把选项键打开或关闭。

实例

将数字键打开，其馀二个灯关闭。

```
# setleds +num -caps -scroll
```

Linux loadkeys命令

Linux loadkeys命令可以根据一个键盘定义表改变 linux 键盘驱动程序转译键盘输入过程。详细的说明请参考 dumpkeys。

语法

```
loadkeys [ -d --default ] [ -h --help ] [ -q --quiet ] [ -v --verbose [ -v --verbose ]...
```

参数:

- -v --verbose：印出详细的资料，你可以重复以增加详细度。
- -q --quiet：不要显示任何讯息。
- -c --clearcompose：清除所有 composite 定义。
- -s --clearstrings：将定串定义表清除。

实例

```
定义按键组合
<pre>
# loadkeys
control alt keycode 88 = F80 //现确定键代码
string F80="w3cschool.cc" //给变变量设定值
//按下 Ctrl + D键 确定输入

//效果：按下 Ctrl +Alt + F12 输出 Lx138.Com

# dumpkeys --funcs-only //显示功能键

……省略部分结果
string F3 = "\033[[C"
string F4 = "\033[[D"
string F5 = "\033[[E"
string F6 = "\033[17~"
string F7 = "\033[18~"
string F8 = "\033[19~"
string F9 = "\033[20~"
string F10 = "\033[21~"
string F11 = "\033[23~"
string F12 = "\033[24~"
string F13 = "\033[25~"
string F14 = "\033[26~"
string F15 = "\033[28~"
string F16 = "\033[29~"
string F17 = "\033[31~"
string F18 = "\033[32~"
string F19 = "\033[33~"
string F20 = "\033[34~"
string Find = "\033[1~"
string Insert = "\033[2~"
string Remove = "\033[3~"
string Select = "\033[4~"
string Prior = "\033[5~"
string Next = "\033[6~"
string Macro = "\033[M"
string Pause = "\033[P"
string F80 = "w3cschool.cc"
```

Linux rdev命令

Linux rdev命令可以用来查询/设置内核映像文件的根设备，RAM 磁盘大小或视频模式。

不带任何参数的 rdev 命令将输出当前根文件系统的 /etc/mtab 文件行。不带任何参数的 ramsize, vidmode, 和 rootflags 将显示帮助信息。

语法

```
rdev [-rsvh ] [-o offset ] [ image [value [ offset ] ] ]</p>
```

但是随著使用者想要设定的参数的不同，底下的方式也是一样：

```
rdev [ -o offset ] [ image [ root_device [ offset ] ] ]
```

```
swapdev [ -o offset ] [ image [ swap_device [ offset ] ] ]
```

```
ramsize [ -o offset ] [ image [ size [ offset ] ] ]
```

```
videomode [ -o offset ] [ image [ mode [ offset ] ] ]
```

```
rootflags [ -o offset ] [ image [ flags [ offset ] ] ]
```

参数：

- -r： 使得 rdev 作为 ramsize 运行。
- -R： 使得 rdev 作为 rootflags 运行。
- -v： 使得 rdev 作为 vidmode 运行。
- -h： 提供帮助。

Linux dumpkeys命令

Linux dumpkeys命令用于显示键盘映射表，输出的内容可以被loadkeys命令识别,改变映射关系。

语法

```
dumpkey[选择参数]
```

参数说明:

- -i 驱动信息(键码范围、数量、状态键)
- -l 详细驱动信息
- -n 十六进制显示
- -f 显示全部信息
- -1 分行显示按键组合
- -S 设定输出格式(0：预设 1：完整 2：分行 3简单)
- --funcs-only 功能键信息
- --keys-only 键组合信息
- --compose-only 普通键信息

实例

显示功能键信息

```
# dumpkeys --funcs-only
string F1 = "\033[[A"
string F2 = "\033[[B"
string F3 = "\033[[C"
string F4 = "\033[[D"
string F5 = "\033[[E"
string F6 = "\033[17~"
string F7 = "\033[18~"
string F8 = "\033[19~"
string F9 = "\033[20~"
string F10 = "\033[21~"
string F11 = "\033[23~"
string F12 = "\033[24~"
string F13 = "\033[25~"
string F14 = "\033[26~"
string F15 = "\033[28~"
string F16 = "\033[29~"
string F17 = "\033[31~"
string F18 = "\033[32~"
string F19 = "\033[33~"
string F20 = "\033[34~"
string Find = "\033[1~"
string Insert = "\033[2~"
string Remove = "\033[3~"
string Select = "\033[4~"
string Prior = "\033[5~"
string Next = "\033[6~"
string Macro = "\033[M"
string Pause = "\033[P"
root@snail-hnlinux:~#
```

显示驱动信息

```
# dumpkeys -i
键值码范围被内核支持: 1 - 255
可绑定到键值的动作最大值: 256
实际使用的键值数: 128
其中 121 已动态分配
被内核支持的动作码值范围
0x0000 - 0x00ff
0x0100 - 0x01ff
0x0200 - 0x0213
0x0300 - 0x0313
0x0400 - 0x0405
0x0500 - 0x05ff
0x0600 - 0x0603
0x0700 - 0x0708
0x0800 - 0x08ff
0x0900 - 0x0919
0x0a00 - 0x0a08
0x0b00 - 0x0bff
0x0c00 - 0x0c08
0x0d00 - 0x0dff
0x0e00 - 0x0e0a
内核支持的功能键数：256
编写定义的最大nr: 256
实际使用的编写定义nr: 68
```

Linux MAKEDEV命令

Linux MAKEDEV命令用于新增 /dev/ 下的装置档案，多数分区已经将所有的档案都产生，故一般而言不太会需要用到这个命令。

语法

```
MAKEDEV -V
MAKEDEV [ -n ] [ -v ] update
MAKEDEV [ -n ] [ -v ] [ -d ] device ...
```

免责声明

W3School提供的内容仅用于培训。我们不保证内容的正确性。通过使用本站内容随之而来的风险与本站无关。W3School简体中文版的所有内容仅供测试，对任何法律问题及风险不承担任何责任。